PHILIPS ADDS NEW DIMENSIONS TO CUSTOMER SERVICE

MVME330 Ethernet Controller P90X0

Field Support Manual

PHILIPS

PHILIPS

MVME330 Ethernet Controller
P90X0

Field Support Manual

A Publication of
PHILIPS TELECOMMUNICATIE EN DATA SYSTEMEN
NEDERLAND B.V.
Customer Service Documentation and Training
Apeldoorn, The Netherlands

Pub. No. 5122 991 35822

Date October 1988

Great care has been taken to ensure that the information contained in this handbook is accurate and complete. Should any errors or omissions be discovered or should any user wish to make a suggestion for improving this handbook, he is invited to send the relevant details to:

PHILIPS TELECOMMUNICATIE EN DATA SYSTEMEN
Customer Service Documentation and Training
P.O. BOX 245
7300 AE APELDOORN
THE NETHERLANDS

STATUS RECORD

TITLE :	MVME330 Ethernet Controller P9070 Field Support Manual
PUBLICATION NUMBER :	5122 991 3582**X**

X	UPD.	SI No.	PAGES AFFECTED	DATE	REMARKS
1			All	8703	Issue date. This manual covers the Motorola User's Manual-MVME330 Ethernet Controller (MVME330/D1- dd 8502)
2			All	8810	Revised version of the Motorola MVME330 manual. The manual is now on level D2

PREFACE:

Unless otherwise specified, all address references are in hexadecimal throughout this manual.

An asterisk (*) following the signal name for signals which are level significant denotes that the signal is true or valid when the signal is low.

An asterisk (*) following the signal name for signals which are edge significant denotes that the actions initiated by that signal occur on high to low transition.

The information in this document about the MVME330 module jumper setting is not explicit for a P90X0 configuration.

To be sure having the correct jumper setting refer to the Customer Engineer Manual P90X0 or to the Installation instructions delivered with the MVME330 board.

WARNING

THIS EQUIPMENT GENERATES, USES, AND CAN RADIATE RADIO FREQUENCY ENERGY AND IF NOT INSTALLED AND USED IN ACCORDANCE WITH THE INSTRUCTIONS MANUAL, MAY CAUSE INTERFERENCE TO RADIO COMMUNICATIONS. IT HAS BEEN TESTED AND FOUND TO COMPLY WITH THE LIMITS FOR A CLASS A COMPUTING DEVICE PURSUANT TO SUBPART J OF PART 15 OF FCC RULES, WHICH ARE DESIGNED TO PROVIDE REASONABLE PROTECTION AGAINST SUCH INTERFERENCE WHEN OPERATED IN A COMMERCIAL ENVIRONMENT. OPERATION OF THIS EQUIPMENT IN A RESIDENTIAL AREA IS LIKELY TO CAUSE INTERFERENCE IN WHICH CASE THE USER, AT HIS OWN EXPENSE, WILL BE REQUIRED TO TAKE WHATEVER MEASURES NECESSARY TO CORRECT THE INTERFERENCE.

Second Edition

Copyright 1988 by Motorola Inc.

First Edition February 1985

SAFETY SUMMARY
SAFETY DEPENDS ON YOU

The following general safety precautions must be observed during all phases of operation, service, and repair of this equipment. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the equipment. Motorola Inc. assumes no liability for the customer's failure to comply with these requirements. The safety precautions listed below represent warnings of certain dangers of which we are aware. You, as the user of the product, should follow these warnings and all other safety precautions necessary for the safe operation of the equipment in your operating environment.

GROUND THE INSTRUMENT.

To minimize shock hazard, the equipment chassis and enclosure must be connected to an electrical ground. The equipment is supplied with a three-conductor ac power cable. The power cable must either be plugged into an approved three-contact electrical outlet or used with a three-contact to two-contact adapter, with the grounding wire (green) firmly connected to an electrical ground (safety ground) at the power outlet. The power jack and mating plug of the power cable meet International Electrotechnical Commission (IEC) safety standards.

DO NOT OPERATE IN AN EXPLOSIVE ATMOSPHERE.

Do not operate the equipment in the presence of flammable gases or fumes. Operation of any electrical equipment in such an environment constitutes a definite safety hazard.

KEEP AWAY FROM LIVE CIRCUITS.

Operating personnel must not remove equipment covers. Only Factory Authorized Service Personnel or other qualified maintenance personnel may remove equipment covers for internal subassembly or component replacement or any internal adjustment. Do not replace components with power cable connected. Under certain conditions, dangerous voltages may exist even with the power cable removed. To avoid injuries, always disconnect power and discharge circuits before touching them.

DO NOT SERVICE OR ADJUST ALONE.

Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.

USE CAUTION WHEN EXPOSING OR HANDLING THE CRT.

Breakage of the Cathode-Ray Tube (CRT) causes a high-velocity scattering of glass fragments (implosion). To prevent CRT implosion, avoid rough handling or jarring of the equipment. Handling of the CRT should be done only by qualified maintenance personnel using approved safety mask and gloves.

DO NOT SUBSTITUTE PARTS OR MODIFY EQUIPMENT.

Because of the danger of introducing additional hazards, do not install substitute parts or perform any unauthorized modification of the equipment. Contact Motorola Field Service Division for service and repair to ensure that safety features are maintained.

DANGEROUS PROCEDURE WARNINGS.

Warnings, such as the example below, precede potentially dangerous procedures throughout this manual. Instructions contained in the warnings must be followed. You should also employ all other safety precautions which you deem necessary for the operation of the equipment in your operating environment.

WARNING

Dangerous voltages, capable of causing death, are present in this equipment. Use extreme caution when handling, testing, and adjusting.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont'd)

TABLE OF CONTENTS (cont'd)

TABLE OF CONTENTS (cont'd)

LIST OF ILLUSTRATIONS

TABLE OF CONTENTS (cont'd)

CHAPTER 1 - GENERAL INFORMATION

1.1 **INTRODUCTION**

This manual provides the following information and instructions for the Motorola MVME330, MVME330-1, and MVME330-2 Ethernet Controller (throughout this manual referred to as MVME330):

- General information
- Hardware preparation and installation instructions
- Software preparation and loading instructions
- Functional description
- Support information

For software interfacing information, refer to the KERNEL user's manual. This manual describes the software protocols and formats necessary to enable the MVME330 to communicate with the Ethernet.

A diagram of the Ethernet architecture and a typical installation is shown in Figure 1-1.

1.2 **FEATURES**

The features of the MVME330 include:

- Microprocessor Unit (MPU):
  10 MHz MC68000 in the MVME330 and MVME330-1
  10 MHz MC68010 in the MVME330-2
- Dynamic memory with parity:
  128Kb in the MVME330
  512Kb in the MVME330-1 and MVME330-2
- Up to 64Kb EPROM (2 sockets)
- VMEbus Requester and VMEbus Master capability
- Node address PROM contains a unique address issued by Xerox Corp
- 2 ms timer interrupts the MPU for protocol software timing
- Am7990/7992 (LANCE/SIA) VLSI
- VMEbus interrupter with programmable vector
- VMEbus to onboard processor interrupter
- Power up self-test
- One Wait State RAM Write Access
- Zero Wait State RAM Read Access
- Compatible with Ethernet 1.0, 2.0 or IEEE 802.3 transceivers

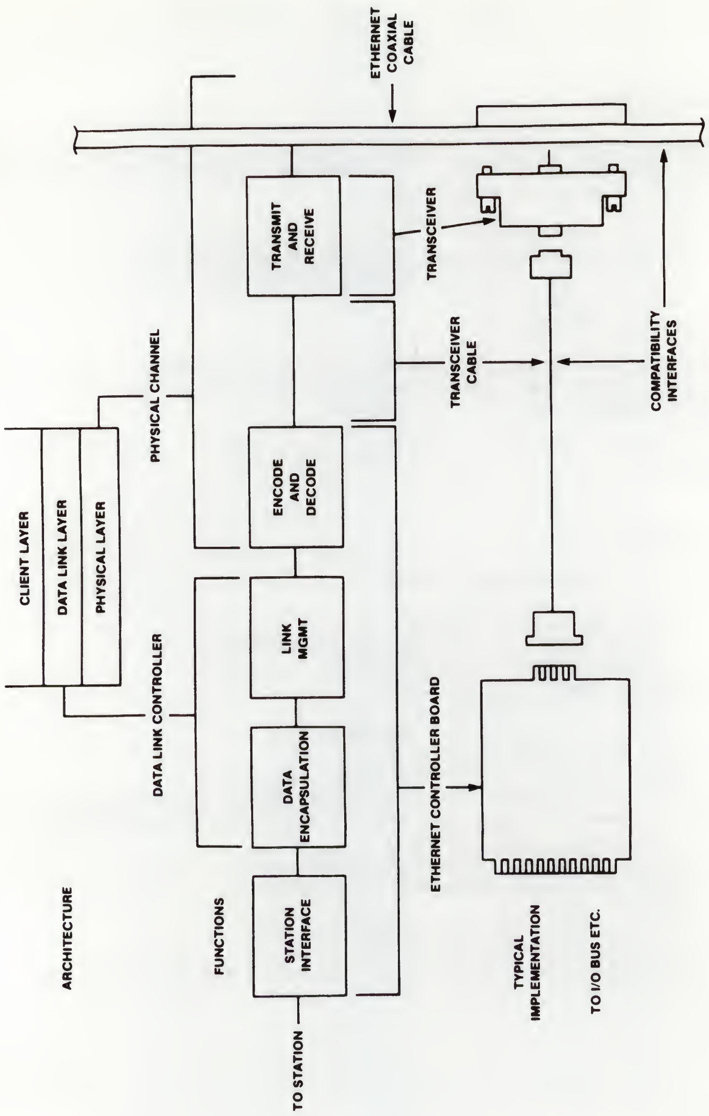


FIGURE 1-1. Ethernet Architecture and Typical Implementation

1.3 SPECIFICATIONS

The MVME330 specifications are given in Table 1-1.

TABLE 1-1. MVME330 Specifications

CHARACTERISTIC	SPECIFICATIONS
Power requirements	4.4 A at +5 Vdc ±5% 0.6 A at +12 Vdc ±5% 0.1 A at -12 Vdc ±5%
Operating temperature	5 degrees C to 50 degrees C @ 25 cfm
Storage temperature	-40 degrees C to 85 degrees C
Humidity	0 to 95% (non-condensing)
Altitude	
Operating	0 to 10,000 feet
Storage	0 to 20,000 feet
Reliability	40,000 hours MTBF
Size	6.3 inches x 9.2 inches (16.0 cm x 23.4 cm)
Connectors	
VMEbus	DIN no. 41612C96
Ethernet transceiver	AMP 745094-1

1.4 GENERAL DESCRIPTION

The MVME330 is a high performance communications processor which provides the physical interface and intelligence necessary to attach information processing devices to Ethernet, a local area network. The MVME330 in conjunction with Ethernet allows a high speed exchange of information.

The MVME330 has a Motorola MC68000 MPU and 128Kb of dynamic RAM. The MVME330-1 has a Motorola MC68000 MPU and 256Kb of dynamic RAM. The MVME330-2 has a Motorola MC68010 MPU and 256Kb of dynamic RAM.

Each MVME330 contains node-specific software and industry standard protocol software (XNS or TCP/IP) for exchanging information throughout the network.

The MVME330 hardware has the following characteristics:

- Two chip Local Area Network Controller for Ethernet (LANCE and SIA)
- MC68000 (MVME330 and MVME330-1) or MC68010 (MVME330-2) MPU that performs supervisory functions over the LANCE
- Closely coupled RAM
- ROM for protocol processing code
- VMEbus interface to a host system
- Ethernet cable interface between the MVME330 transceiver connector and the Ethernet transceiver equipment
- Can act as either system bus master or slave, allowing the MVME330 to write to and read from bus memory
- Timing and signaling utilities required to maintain a communication environment
- Installs in the user's system chassis card rack and connects to the chassis backplane
- Periodic hardware generated interrupt

The hardware generated interrupt notifies the software that it is time to perform the software memory refresh. This interrupt occurs about every two milliseconds. The software memory refresh results in approximately five percent overhead.

The MVME330 software reference manual describes the related protocols that are required for MVME330 operation. The MVME330 specific software has the following characteristics:

- Communication between the VMEbus and the MVME330 is handled by the Bus Interface Protocol (BIP) software.
- Communication between the MVME330 and the network's physical medium (the coaxial cable) is handled by the MVME330 kernel software and the LANCE devices.
- All MVME330 protocols are compatible with Ethernet 1.0, Ethernet 2.0, and IEEE 802.3 specifications.

1.5 VME CONFIGURATION

The VME configuration is as follows:

1.5.1 Master Data Transfer

A24:D16
TOUT = 8 or 16 microseconds
ADDRESS MODIFIERS = 3E, 3D, 3A, 39, 2D, 29

1.5.2 Slave Data Transfer

A24:D16
ADDRESS MODIFIERS = 3D, 39

1.5.3 Requester

Any one of: R(0), R(1), R(2), or R(3), (STAT), RWD (Release after each MPU VMEbus cycle).

1.5.4 Interrupter Options

Any one of I(1), I(2), I(3), I(4), I(5), I(6), I(7), (STAT)

1.6 MEMORY CONSIDERATIONS

The LANCE device on the MVME330 requires two special memory considerations: location of the memory buffers, and possible bus time-out.

1.6.1 Location of Memory Buffers

Unlike most VMEbus system devices, the LANCE memory buffers must reside on the MVME330 rather than in system memory. The reason for this is that LANCE uses eight-word bursts when writing received Ethernet data to memory and when reading data from memory to send on the Ethernet. If the memory buffer used was system memory, excessive intermittent delays would be experienced by other devices attempting to access system memory.

Ethernet data should be staged in a memory other than the LANCE SILO register between Ethernet transfers and system bus transfers. The transfer of the data between MVME330 memory and user memory can be managed by either the MVME330 or the host processor. In Motorola MVME330 software, all transfers are managed by the MVME330 to provide an efficient data transfer.

1.6.2 Access Delay

During LANCE activity, the host can incur up to a nine microsecond wait before being granted MVME330 memory access. The nine microsecond delay is because, as indicated above, the LANCE performs transfers in eight-word bursts (when LANCE acquires the local data bus it holds onto the bus until all eight words are transferred).

If nine microseconds exceeds a host's bus time-out interval, the driver should place the mailboxes in host memory and have the MVME330 perform all data transfers between the MVME330 memory and system bus memory. Motorola software places the mailboxes in MVME330 memory. Therefore the system resource time-out and any VMEbus master accessing the MVME330 memory must have a bus time-out setting of 16 microseconds or greater.

1.7 RELATED DOCUMENTATION

The following manuals are applicable to the MVME330. If these manuals are not shipped with this product, they may be purchased from Motorola's Literature Distribution Center, 616 West 24th Street, Tempe, AZ 85282; telephone (602) 994-6561.

DOCUMENT TITLE	MOTOROLA PUBLICATION NUMBER
VMEbus Specification Manual	HB212/D
MVME330 LAN Controller Kernel Software Reference Manual	MVME330KSW
CMC XNS User's Manual (VERSAdos)	MVMEXNSVDOS
CMC XNS User's Manual (SYSTEM V/68)	MVMEXNSV68

NOTE: Although not shown in the above list, each Motorola MCD manual publication number is suffixed with characters which represent the revision level of the document, such as "/D2" (the second revision of a manual); supplement bears the same number as the manual but has a suffix such as "/A1" (the first supplement to the manual).

1.8 MANUAL TERMINOLOGY

Throughout this manual, a convention has been maintained whereby data and address parameters are preceded by a character which specifies the numeric format as follows:

$	dollar	specifies a hexadecimal number
%	percent	specifies a binary number
&	ampersand	specifies a decimal number

Unless otherwise specified, all address references are in hexadecimal throughout this manual.

An asterisk (*) following the signal name for signals which are level significant denotes that the signal is true or valid when the signal is low.

An asterisk (*) following the signal name for signals which are edge significant denotes that the actions initiated by that signal occur on a high to low transition.

In this manual, activation and deactivation are used to specify forcing a signal to a particular state. In particular, activation and activate refer to a signal that is active or true; deactivation and deactivate indicate a signal that is inactive or false. These terms are used independently of the voltage level (high or low) that they represent.

CHAPTER 2 - HARDWARE PREPARATION AND INSTALLATION

2.1 INTRODUCTION

This chapter provides hardware preparation and installation instructions for the MVME330.

2.2 HARDWARE PREPARATION

2.2.1 General Information

CAUTION

AVOID TOUCHING AREAS OF INTEGRATED CIRCUITRY; STATIC DISCHARGE CAN DAMAGE CIRCUITS.

2.2.1.1 Headers

A list of headers and their functions is given in Table 2-1. See Figure 2-1 for header locations.

TABLE 2-1. Headers

HEADER	DESCRIPTION
JP01	Alternate Transceiver Connection
JP02	PROM DTACK Timing Select
JP03	PROM Control
JP04	Resource Time-out Select
JP05	Board Address Select
JP07	Ethernet Type Select
JP08	Interrupt Acknowledge Level Select
JP09	Interrupt Request Priority Level Select
JP10	Bus Request Priority Level Select
JP11	System Clock Disable
JP12	Ethernet Shield Grounding Option

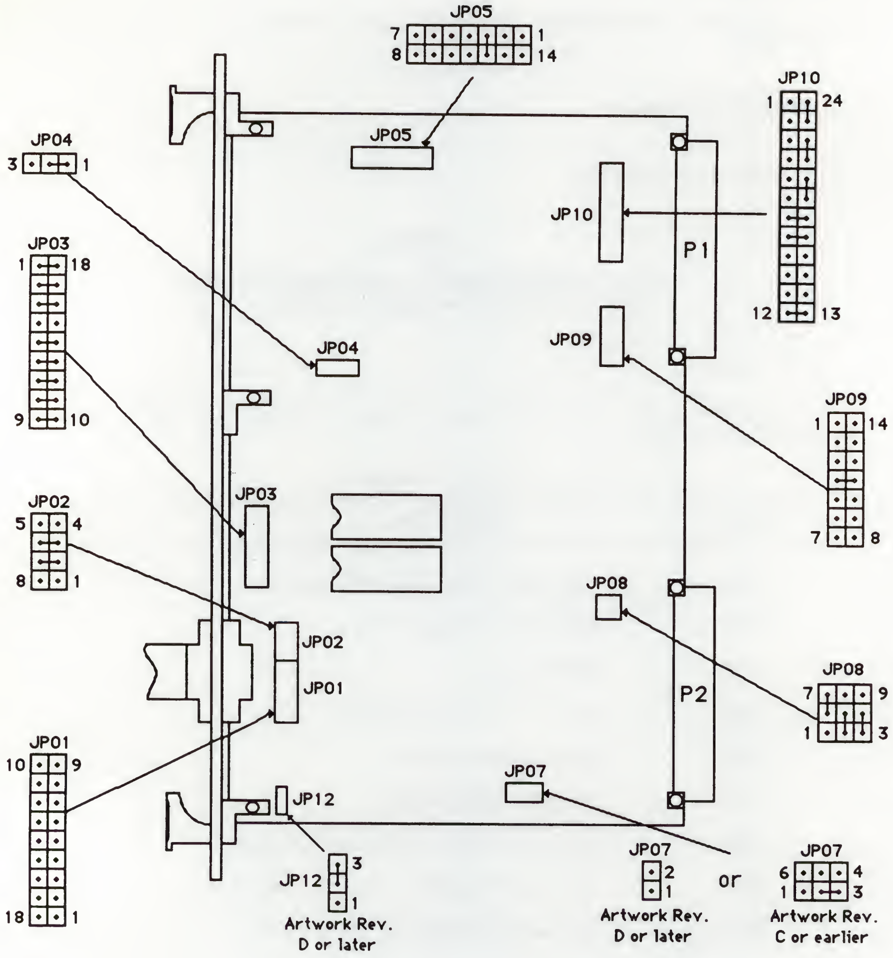


FIGURE 2-1. MVME330 Header Locations and Standard Configurations

2.2.1.2 Standard Header Configurations

The standard factory header configurations are listed in Table 2-2. See Figure 2-1 for header locations.

TABLE 2-2. Standard Header Configurations

HEADER	FACTORY JUMPER CONFIGURATION	
JP01	None	
JP02	2-7	3-6
JP03	1-18	6-13
	2-17	7-12
	3-16	8-11
	4-15	9-10
JP04	1-2	
JP05	3-12	
JP06	Not user configurable	
JP07	2-3 (revision C)	
	none: revision D and later	
JP08	2-5	
	3-6	
	4-7	
JP09	4-11	
JP10	7-18	19-20
	8-17	21-22
	12-13	23-24
JP11	Not user configurable	
JP12	2-3	

2.2.1.3 Ethernet Interface Capabilities

MVME330s of revision levels C and above will interface with transceivers meeting IEEE 802.3, Ethernet 2.0, or Ethernet 1.0 specifications. (The revision level refers to the matrix revision, not the PCB revision. The matrix revision level is located to the right of, or as a suffix to, the 01-W*xxxx*B*xx* number on the module).

2.2.1.4 IEEE 802.3 and Ethernet 2.0 Configuration

The specifications for IEEE 802.3 (no dc voltage offset and no dc differential voltage at quiescence) and Ethernet 2.0 (dc voltage offset of 0 to 5 volts and no dc differential) allow the same configuration to be used for both transceivers. The MVME330 is shipped in the IEEE 802.3/Ethernet 2.0 configuration.

2.2.1.5 Ethernet 1.0 Configuration

The specifications for Ethernet 1.0 are: dc voltage offset of 0 to 5 volts, and a differential voltage of 700 millivolts at quiescence. This requires the MVME330 to be reconfigured.

Header JP07 and socket U1 (shipped with the transformer installed) are reconfigured to allow Ethernet 1.0 operation. Refer to Paragraph 2.2.2.7 for specific jumpering information.

2.2.2 Header Descriptions

2.2.2.1 JP01: Alternate Transceiver Connection

If connector P2 of the VMEbus is used to route the signals of the Ethernet transceiver connection, all jumpers on JP01 should be installed. In this case, the transceiver cable will originate from a separate connection instead of the MVME330. When installed, the jumpers connect opposing members of the two rows (i.e., 1-18, 2-17, etc.). Header JP01 is illustrated as follows.

NOTE

Do not use J2 connector if JP01 is used.

```
  P|   P|   P|   P|   P|   P|   P|   P|   P|
  G|   2|   2|   2|   2|   2|   2|   2|   2|
  3|   C|   C|   T|   T|   R|   R|   2|   2|
  0|   P|   M|   P|   M|   P|   M|   2|   2|
+--------------------------------------------+
|  18   17   16   15   14   13   12   11   10 |
|   o    o    o    o    o    o    o    o    o |
|                                            |        JP01
|   o    o    o    o    o    o    o    o    o |
|   1    2    3    4    5    6    7    8    9 |
+--------------------------------------------+
  G|   C|   C|   T|   T|   R|   R|   +|   G|
  N|   P|   M|   P|   M|   P|   M|   1|   N|
  D|   I|   I|   I|   I|   I|   I|   2|   D|
   |    |    |    |    |    |    |   F|    |
```

2.2.2.2 JP02: PROM DTACK Timing Select

PROM DTACK timing is selected according to the speed of the PROMs used. Factory configuration is for no wait states (JP02 2 to 7, and 3 to 6). Table 2-3 lists JP02 jumpering and the number of wait states.

```
                 R|         R|
                 O|         O|
                 M|         M|
                 D+---+   +--A+
                  |   |   |  C|
                  |   |   |  K|
                  |   |   |  -|
                +----------------+
                | 8   7   6  5   |
                | o   o   o  o   |
      JP02      |     |   |      |
                | o   o   o  o   |
                | 1   2   3  4   |
                +----------------+
                 R|  R|  M| R|
                 M|  M|  A| O|
                 D|  D|  S| M|
                 4|  2|  -| D|
                  |   |   | -|
```

TABLE 2-3. Number of Wait States

NUMBER OF WAIT STATES	JUMPER ON JP02	PROM ACCESS TIME
0	2-7 and 3-6	200 ns maximum
2	2-7 and 4-5	400 ns maximum
4	1-8 and 4-5	600 ns maximum

2.2.2.3 JP03: PROM Control

Header JP03 allows selection of a variety of 24-pin or 28-pin PROMs. Factory configuration is for the Am2764 or Am27128-2. Refer to the specification for the PROM to be used for a description of these signals. Table 2-4 provides a description of the jumper outputs (JP03 pin numbers). Header JP03 is illustrated below.

To use the Am2756-2 PROM, remove the shunt between pins 9-10 and install a jumper between pins 1-18, 2-17, 3-16, 4-15, 5-14, 6-13, 7-12, and 8-11.

```
                     P|  P|  P|  P|  P|  P|  P|  P|  P|
                     2|  2|  2|  2|  2|  2|  2|  2|  1|
                     7|  0|  2|  3|  6|  7|  8|   |   |
                    +------------------------------------+
                    | 9   8   7   6   5   4   3   2   1  |
                    | o   o   o   o   o   o   o   o   o  |
         JP03       | |   |   |   |   |       |   |   |  |
                    | o   o   o   o   o   o   o   o   o  |
                    | 10  11  12  13  14  15  16  17  18 |
                    +------------------------------------+
                     V|  G|  R|  A|  A|  A|  V|  A|  V|
                     C|  N|  O|  1|  1|  1|  C|  1|  C|
                     C|  D|  M|  2|  4|  5|  C|  3|  C|
                      |   |  E|   |   |   |   |   |   |
                      |   |  -|   |   |   |   |   |   |
```

TABLE 2-4. Signal Table

JP03 JUMPER OUTPUT (PIN #)	PROM (27128-2) CONTROL SIGNAL
1	VCC
2	A13
3	VCC
4	PGM (Program Enable)
5	A14
6	A12
7	OE (Output Enable)
8	CE (Chip Enable)
9	PGM (Program Enable)

2.2.2.4 JP04: Resource Time-Out Select

This header allows selection of either an 8- or 16-microsecond (+0, -2 microsecond) Resource Time-Out (RTO). (The module is factory-configured for an 16-microsecond RTO by jumpering pin 1 to pin 2.) The resource time-out select header is illustrated as follows.

```
              JP04
            +-----+
            |     |
     RTO8   |3 o  |
            |     |
            |2 o--+-----RTO
            |  |  |
     RTO16  |1 o  |
            |     |
            +-----+
```

2.2.2.5 JP05: Board Address Select

User selection of PC board address. AJ23-17 are compared with VME address bits A23-17 to determine the modulo 128Kb slave response window. Installed jumpers are binary 0 while a jumper not installed is a binary 1. JP05 is illustrated as follows.

```
                           JP05
         +----------+                      +----------+
AJ17-----+8-o   o-7-+----+        AJ17-----+8-o   o-7-+----+
         |          |    |                 |          |    |
AJ18-----+9-o   o-6-+----+        AJ18-----+9-o---o-6-+----+
         |          |    |                 |          |    |
AJ19-----+10o   o-5-+----+        AJ19-----+10o   o-5-+----+
         |          |    |                 |          |    |
AJ20-----+11o   o-4-+----+        AJ20-----+11o   o-4-+----+
         |          |    |                 |          |    |
AJ21-----+12o---o-3-+----+        AJ21-----+12o---o-3-+----+
         |          |    |                 |          |    |
AJ22-----+13o   o-2-+----+        AJ22-----+13o---o-2-+----+
         |          |    |                 |          |    |
AJ23-----+14o   o-1-+----+----GND AJ23-----+14o---o-1-+----+----GND
         +----------+                      +----------+

$DE0000 (Factory Configuration)            $1A0000
```

Jumper absent: AJxx = 1
Jumper installed: AJxx = 0
Example:

For DE0000, jumper pins 3 to 12 (factory configuration as shown above)

For 1A0000, jumper pins 1 to 14, 2 to 13, 3 to 12, 6 to 9 (shown above)

2.2.2.6 JP06: Not Used

2.2.2.7 JP07: Ethernet Type Selection

The MVME330, MVME330-1, and MVME330-2 are fully tested with transceivers manufactured by TCL Incorporated, Fremont, California. The Rev. C and later versions of the MVME330 offer the capability of interfacing with either an Ethernet 2.0/IEEE 802.3 compatible transceiver or an Ethernet version 1.0 transceiver. The TCL part number for the Ethernet 2.0/ IEEE 802.3 transceiver is 2010I. The TCL part number for the Ethernet version 1.0 transceiver is 2010EB.

JP07 for the Rev C MVME330 module is jumpered differently for the two transceiver types (illustrated below). For Ethernet 2.0/IEEE 802.3 transceivers, jumper JP07 pins 2-3 and leave transformer U1 installed. For Ethernet version 1.0 transceivers, jumper JP07 pins 1-6 and 2-3, remove the transformer at U1 and install a jumper platform (Samtec part number LVP-316-G or equivalent) at U1 with the following jumpers: 1-16, 2-15, 3-14, 4-13, 5-12, 6-11, 7-10 and 8-9.

```
                    REVISION "C" AND EARLIER

       JP07              JP07                       U1
   +---------+       +---------+             +----------+
   |         |       |         |             |1 o---o 16|
   |1 o   o 6|       |1 o---o 6|             |          |
   |         |       |         |             |2 o---o 15|
   |2 o   o 5|       |2 o   o 5|             |          |
   |  |      |       |  |      |             |3 o---o 14|
   |3 o   o 4|       |3 o   o 4|             |          |
  -|         |       |         |             |4 o---o 13|
   +---------+       +---------+             |          |
   IEEE 802.3        Ethernet 1.0            |5 o---o 12|
       or            also, remove            |          |
   Ethernet 2.0      the trans-  ----------->|6 o---o 11|
                     former at U1            |          |
                     and install a           |7 o---o 10|
                     jumper plat-            |          |
                     form wire as            |8 o---o  9|
                     illustrated.            +----------+
```

For Rev. D and later versions of MVME330, JP07 is configured with only pin 1 and pin 2. No jumper is installed for Ethernet 2.0 or IEEE 802.3. For Ethernet 1.0, install a jumper between pin 1 and pin 2, remove the transformer at U1 and replace it with a jumper platform (Samtec part number LVP-316-G or equivalent) with the following jumpers: 1-16, 2-15, 3-14, 4-13, 5-12, 6-11, 7-10 and 8-9.

```
                 REVISION  D
                  AND LATER
                    JP07                          U1
        +------------------+               +----------+
        |         |        |               |          |
TSEL----|    1    |    2   |------GND      |1 o---o 16|
        |         |        |               |          |
        +------------------+               |2 o---o 15|
        Install jumper  JP07 for           |          |
        Ethernet 1.0 only.                 |3 o---o 14|
        Also, remove  the trans-           |          |
        former at U1 and install---------->|4 o---o 13|
        a jumper  platform wired           |          |
        as illustrated.                    |5 o---o 12|
                                           |          |
                                           |6 o---o 11|
                                           |          |
                                           |7 o---o 10|
                                           |          |
                                           |8 o---o  9|
                                           +----------+
```

2.2.2.8 JP08: Interrupt Acknowledge Level Select

This header, in conjunction with header JP09, allows user selection of the VMEbus interrupt level. JP08 is used to select Interrupt Acknowledge (IACK) Levels 1 through 7. The level selected must agree with the interrupt request level configured with JP09.

The three JP08 output lines (VJ1 through VJ3) are jumpered to either GND or +5 Vdc to give 3-bit positive binary code for the IACK level; VJ3 is the most significant bit. JP08 is illustrated as follows.

```
                 JP08

        +------------+
GND     |            |   VCC
--------+-o   o---o--|------------
     |  | 1   |4  7  |   VJ3
     |  |     +------|------------
     |  |            |   VCC
     |--+-o---o   o--|------------
     |  | 2   |5  8  |   VJ2
     |  |     +------|------------
     |  |            |   VCC
     |--+-o---o   o--|------------
        | 3   |6  9  |   VJ1
        |     +------|------------
        |            |
        +------------+
```

Example:

IACK level 4 (factory configuration) is selected by jumpering to the following pin pairs.

```
GND      VJn     VCC

1        4------7
2--------5      8
3--------6      9
```

2.2.2.9 JP09: Interrupt Request Priority Level Select

This header is used to route the onboard Interrupt Request (IRQ*) signal to one of the seven VMEbus interrupt request lines, IRQ1* through IRQ7*.

The level selected must agree with the interrupt acknowledge priority level configured with JP08. Header JP09 with factory configuration for level 4 is illustrated as follows.

```
                  I|  I|  I|  I|  I|  I|  I|
                  R|  R|  R|  R|  R|  R|  R|
                  Q|  Q|  Q|  Q|  Q|  Q|  Q|
                  1|  2|  3|  4|  5|  6|  7|
                  *|  *|  *|  *|  *|  *|  *|
                +-----------------------------+
                | 7   6   5   4   3   2   1   |
                | o   o   o   o   o   o   o   |
    JP09        |             |               |
                | o   o   o   o   o   o   o   |
                | 8   9   10  11  12  13  14  |
                +-----------------------------+
                  |   |   |   |   |   |   |
                  +---+---+---+---+---+---+
                                          I|
                                          R|
                                          Q|
                                          *|
```

2.2.2.10 **JP10: Bus Request Priority Level Select**

This header allows user selection of the bus request priority, which may have the values 0, 1, 2, or 3. When a priority level *n* is chosen, the following jumpers must be installed.

a. The onboard Bus Request (BRX*) signal is tied to the VMEbus Bus Request (BRn*) signal of the chosen level, either BR0*, BR1*, BR2*, or BR3*.

b. The onboard Bus Grant Out (BGO*) signal is tied to the outgoing BGnOUT* signal.

c. The incoming BGnIN* signal is tied to signal BGI*.

d. For the three levels not chosen, the VMEbus BGxIN* signal must be tied to the corresponding BGxOUT* signal.

Header JP10 with factory configuration for BR3*/BG3* is illustrated as follows.

Example:

For BR3/BG3 (factory configuration)

```
Jumpers on JP10 are:     7-18
                         8-17
                        12-13
                        19-20
                        21-22
                        23-24
```

```
              B|              B|   --------.-------.-------|
              R|---.---.---   G|   |       |       |      B|
              X|   |   |   |  O|   |---.---|---.---|---   G|
              *|   |   |   |   |   |   |   |   |   |   |  I|
               |   |   |   |   |   |   |   |   |   |   |  *|
             +-----------------------------------------------+
             | 12  11  10  9   8   7   6   5   4   3   2   1 |
             | o   o   o   o   o   o   o   o   o   o   o   o |
    JP10     | |               |   |                         |
             | o   o   o   o   o   o   o   o   o---o   o---o |
             | 13  14  15  16  17  18  19  20  21  22  23  24|
             +-----------------------------------------------+
               |   |   |   |   |   |   |   |   |   |   |   |
              B|  B|  B|  B|  B|  B|  B|  B|  B|  B|  B|  B|
              R|  R|  R|  R|  G|  G|  G|  G|  G|  G|  G|  G|
              3|  2|  1|  0|  3|  3|  2|  2|  1|  1|  0|  0|
              *|  *|  *|  *|  O|  I|  O|  I|  O|  I|  O|  I|
               |   |   |   |  U|  N|  U|  N|  U|  N|  U|  N|
               |   |   |   |  T|  *|  T|  *|  T|  *|  T|  *|
               |   |   |   |  *|   |  *|   |  *|   |       |
```

2.2.2.11 JP11: System Clock Disable

MVME330's with serial numbers between 600 and 1198 have a jumper in JP11. The jumper must be in to connect the 20 MHz crystal oscillator to the clock. The jumper has been replaced with hard wiring for later versions.

2.2.2.12 JP12: Ethernet Shield Grounding

JP12 allows selection of chassis or logic ground for pin 1 of the transceiver cable connector (J2). The factory setting is pins 2 to 3, connecting JP2 pin 1 to logic ground. This setting is recommended for Ethernet 2.0 or IEEE 802.3 transceiver installations. Ethernet 1.0 transceiver installations usually require pin 1 of J2 to be connected to chassis ground (JP12 pins 1 and 2).

```
                                     JP12
                                +--------------+
          chassis GND           |    |    |    |
     (metallic front plate)---->| 1  | 2--|--3 |<--Logic GND
                                |    |    |    |
                                +--------------+
                                       |
                                       |
                                   J2-Pin 1
                  Setting for IEEE 802.3/Ethernet 2.0 shown
```

2.3 INSTALLATION INSTRUCTIONS

2.3.1 Installing the Ethernet Cable

Required items:

. Ethernet cable
. Barrel connectors to join sections of cable
. 2 terminators
. 1 cable ground clamp
. 1 ground wire of appropriate length (AWG rating 2 - 8, insulated)
. Wire strippers
. Electrical tape

Install the Ethernet cable as follows.

a. Plan the network layout.

NOTE

The distance between node taps must be an integral multiple of 2-1/2 meters (8.2 feet). The bulk cable is marked with a heavy black bar every 2-1/2 meters for handy reference.

b. Build the cable, using barrel connectors as necessary. If working with sections of cable, each of these sections is terminated by a male terminator. Two sections of cable can be put together with a barrel connector. Remove the terminator cap to accept the barrel connector.

NOTE

Total cable length may not exceed 500 meters (1640 feet).

c. Each free end of the cable must be capped with a terminator.

d. Route the cable per network design.

e. Ground the cable.

CAUTION

THE CABLE MAY ONLY BE GROUNDED AT ONE POINT.

NOTE

One of the terminators is often an easily accessible grounding point.

Terminate the ground at any existing ground termination point, or at a separately driven ground rod. Place a cable ground clamp around the connector chosen as the ground origin. Be careful not to overtighten the screws securing the clamp, or damage to the connector may result.

1. Cut the ground wire to the appropriate length.
2. Strip 1/2-inch of the insulation from each end of the ground wire; insert one end of the wire in the receptacle in the ground clamp.
3. Tighten the screw holding the ground wire in place.
4. Attach the free end of the ground wire to the ground termination point.

f. Wrap all connector junctions and terminators with electrical tape.

2.3.2 Attaching the Transceiver

Required items:

. 9/16-inch open-end wrench
. Cable coring tool
. Shield removing tool
. Insulation piercing tool
. Razor knife
. Tap blocks (one per network node)
. Transceivers (one per network node)
. Transceiver cables (one per network node)

NOTE

The following steps must be performed on each network node.

a. The tap block and the transceiver are shipped together in order to protect the transceiver's "stinger". Unscrew the transceiver from the tap block.

CAUTION

Handle the transceiver "stinger" with care.

b. The tap blocks must be installed at integer multiples of 2-1/2 meters. The Ethernet cable is marked every 2-1/2 meters to indicate where a transceiver may be located. Clamp the tap block onto the cable at the mark nearest to the node being installed.

The threaded hole in the tap block should point toward the node being installed. Use the 9/16-inch open-end wrench to tighten the tap block on the cable.

c. Insert the cable coring tool into the threaded hole in the tap block. Screw the tool in until the threads bottom out, then work it back and forth a few times. This removes the insulator jacket from the cable. (It may also remove some of the braided shielding layer below; this is normal.) Remove the cable coring tool.

d. Insert the shield removing tool into the threaded hole in the tap block. This tool is not threaded. Work it back and forth a few times to remove the remaining braided shielding from the cable.

e. Use the razor knife to dig out any remaining shielding from the hole in the tap block.

CAUTION

IF ANY BRAIDED SHIELDING REMAINS IN THE HOLE, THE TRANSCEIVER CAN SHORT THE COAXIAL CABLE AND CRASH THE NETWORK.

f. Insert the inner insulation piercing tool into the threaded hole in the tap block. Screw the tool down until the threads bottom out, then unscrew and remove it. This drills a tiny hole in the cable.

g. If not already done, install an "O" ring onto the transceiver threads.

CAUTION

HANDLE THE STINGER WITH CARE.

h. Insert the transceiver threads into the threaded hole in the tap block, and screw it down until finger tight.

CAUTION

DO NOT OVERTIGHTEN THE TRANSCEIVER.

i. Connect the transceiver cable to the transceiver. Be sure the locking plate is in the closed position. Extend the cable to the node being installed, according to your network plan.

The user may wish to anchor the transceiver and cable to an immobile object for support. If this is done, leave some slack in the cable to prevent the transceiver cable from pulling and possibly damaging the Ethernet cable.

2.3.3 Installing the MVME330 Module

a. Power down the system.

CAUTION

INSERTING OR REMOVING MODULES WHILE POWER IS APPLIED COULD RESULT IN DAMAGE TO MODULE COMPONENTS.

b. The MVME330 module may be installed in any unused double height card slot.

c. Carefully slide the MVME330 in the card slot. Fasten the module in the chassis with the screws provided. Make good contact with the mounting rails to minimize RFI emissions.

d. Power up the system. The green RUN LED should illuminate when the host is operating.

CHAPTER 3 - OPERATING INSTRUCTIONS

3.1 INTRODUCTION

This section provides an overview of hardware operation.

3.2 CONTROLS AND INDICATORS

The MVME330 has a FAIL indicator and a RUN indicator. Both indicators are controlled in part by the diagnostic self test. Refer to Chapter 4 for information on the diagnostics self test.

The MVME330 has no controls.

3.2.1 RUN Indicator

The green RUN LED lights when the diagnostic self tests are successfully completed. It will remain on unless a parity error interrupt occurs. If a parity error interrupt occurs, processing will continue but the RUN LED will extinguish and remain off until the MVME330 is reset or restarted.

3.2.2 FAIL Indicator

The red FAIL LED lights at power on reset and remains lit until the diagnostic self tests are successfully completed.

3.3 OPERATION

For information on the specific protocol in use, refer to the CMC XNS User's Manual, Motorola publication MVMEXNSV68.

CHAPTER 4 - FUNCTIONAL DESCRIPTION

4.1 INTRODUCTION

This section provides an overall block diagram and operational descriptions for the MVME330.

4.2 DESCRIPTION

The block diagram is shown in Figure 4-1. The module operates through the following functional blocks.

4.2.1 MC68000/MC68010 MPU

The processor on board MVME330 and MVME330-1 is a 10 MHz MC68000 Microprocessor Unit (MPU). The processor on board MVME330-2 is a 10 MHz MC68010. See Figure 5-4 for the MVME330 Memory Map.

Processor duties include:

. Moving commands and data to and from system memory.

. Generating bus interrupts.

. Executing the network and transport XNS protocol layers.

. Providing timer functions.

. Controlling the LANCE in its execution of the data link protocol under direction of the communications executive.

4.2.2 Node Address PROM

Every Ethernet station (host and slave) has been assigned a worldwide 48-bit address by the Ethernet Address Administration Office. This address resides in the Node Address (512 X 4-bit) PROM and is read by the application protocol software during the negotiation procedure. The address is used for station identification on all data transmissions.

4.2.3 EPROM Sockets

The MVME330 has two 28-pin sockets for EPROM. The sockets accept the following size EPROMs of various access times: 32K x 8, 16K x 8, 8K x 8, and 4K x 8.

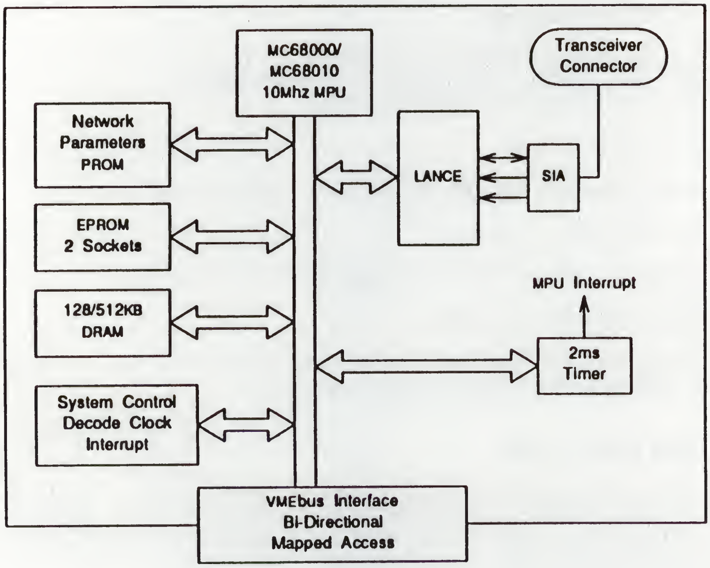


FIGURE 4-1. MVME330 Block Diagram

4.2.4 RAM

MVME330 has 128Kb of dynamic memory with parity. The MVME330-1 and MVME330-2 have 512Kb of dynamic memory with parity. The memory causes no wait states when accessed from the onboard MPU. Software refresh occupies 5.0 percent of the MPU bandwidth.

Parity error is reported as an interrupt.

The dynamic memory is accessible from the system bus, from LANCE, and from the MPU. The VMEbus has highest access priority, followed by the LANCE and MPU with lowest priority. Refer to Chapter 1 for a discussion on memory considerations.

4.2.5 Local Bus Interface

A single onboard address and data bus is under control of the MPU. The bus is shared between:

. LANCE
. RAM
. ROM
. System Bus (VMEbus) Interface
. MPU

Bus requests to the MPU cause it to yield the local bus for VMEbus to MVME330 access and LANCE Direct Memory Access (DMA) to onboard memory.

While one resource is using the local bus, all other resources are inactive. For example: VMEbus access to onboard memory and MPU operation could be held off for more than nine microseconds while LANCE completed eight DMA cycles.

Even during periods of high data transfer, the MPU can have up to 88 percent of the local bus bandwidth. At 120 packets/sec with 1500 bytes per packet, and an average of 800 nsecs per word DMA access, the LANCE DMA will take approximately 7.4 percent of the local bus bandwidth. With 5 percent of the bus used for software refresh, and negligible VMEbus access, this leaves approximately 88 percent of the local bus for the MPU.

LANCE DMA use of the local bus is sporadic, occupying a large percentage of the bus when in use. During the active reception or transmission of data packets on the 10 megabit Ethernet, LANCE alone uses more than 50 percent of the local bus bandwidth.

The size of the local memory has a profound effect on the operation of the MVME330. To prevent loss of status and data the number of back-to-back Ethernet packets that can be processed by the protocol before the host processes the data is limited. The MVME330 can buffer as many back-to-back Ethernet packets as there is receive data buffer space available.

The MPU, not the host, moves data on and off the MVME330. The host processor tells the MVME330 where it wants the data, and the MPU moves it there. If the host processor moved the data directly it could produce inefficiencies by conflicting with LANCE DMA to the onboard memory.

4.2.6 Clock Timer

A 2 ms timer causes a periodic interrupt to the onboard MPU for protocol processing software timers and software refresh.

4.2.7 Local Area Network Controller for Ethernet (LANCE) Am7990

The LANCE has the following features.

. Ethernet 10 megabit data rate
. MC68000/MC68010 MPU-compatible interface
. 16-bit data bus
. Multiplexed address/data bus
. DMA controller with 24-bit addressing
. Sophisticated buffer management structure
. 48-byte SILO data buffer
. Diagnostic aids
. Three modes of receive addressing comparisons
. CSMA/CD network access algorithm
. Extensive error reporting

4.2.7.1 SILO Buffer

The 48 bytes of internal SILO buffer considerably increase the allowed initial response time for a DMA transfer request. Once the DMA has started, an average of one word must be transferred every 1.6 microseconds to sustain the 10 megabit data rate.

4.2.7.2 MC68000/MC68018 Compatable Interface

LANCE readily interfaces with the MC68000/MC68010 MPU. A complication of interfacing with the MPU is a multiplexed address/data bus, which requires external latches for these lines. LANCE implements the signals for controlling these latches.

4.2.7.3 Onchip DMA Controller

The onchip DMA controller places LANCE in the high performance category of Ethernet controllers.

4.2.7.4 Available Diagnostics

Available diagnostics include internal or external loopback tests, CRC logic check, and data path validation.

4.2.7.5 Address Comparison Modes

The three modes of network receive address comparison are:

- Physical mode, which does a full comparison of all 48 destination address bits.
- Logical mode, which puts the 48 bits through a hash filter to determine 1 of 64 logical types called multicast mode.
- Promiscuous mode, which receives all packets, regardless of address.

4.2.7.6 CSMA/CD Algorithm

LANCE implements the full CSMA/CD network access algorithm. Upon detection of a collision, it sends a jam signal and then follows a back-off algorithm before attempting to transmit again. After 16 successive collisions, LANCE reports an error.

4.2.7.7 Error Reporting

The following errors cause an interrupt to be generated.

- 16 successive collisions (see above)
- Babbling transmitter (transmission of more than 1518 bytes)
- Collision detection circuitry nonfunctional
- Missed packet due to insufficient buffer space
- Memory time-out (DMA not complete in a reasonable amount of time)

Individual packet errors include

- CRC error
- Framing error
- SILO overflow or underflow error

4.2.7.8 Buffer Management

LANCE controls an "own" bit for each buffer which signals when the buffer has been filled or emptied and is available for action by the processor.

Buffer management features include:

- A defined circular queue of buffer descriptors called descriptor rings.
- Up to 128 buffers may be queued awaiting processing by LANCE at any one time.
- Data buffers may be chained to handle long packets in multiple data buffer areas.

4.2.8 Serial Interface Adapter (SIA)

The SIA performs the Manchester encoding and decoding necessary for interfacing LANCE to Ethernet. It is compatible with standard Ethernet bus transceivers operating at 10 megabits/sec.

The decoder acquires the clock and data within six bit times (600 nsec). It features guaranteed carrier detection and collision detection threshold limits and transient noise rejection.

The receiver decodes Manchester data with up to plus or minus 20 nsec clock jitter, which represents 1/5 of a bit time.

IEEE 802.3 electrical requirements are met with the SIA to transceiver interface.

4.2.9 Transceiver Connector

The transceiver connection on the MVME330 is a 15-pin conductor connection (MIL-C-24308) that connects the MVME330 to the transceiver cable. The connector is female with a slide latch assembly. The transceiver connector meets applicable IEEE 802.3 specifications.

4.2.10 Bus Requester

The bus requester and the controller as bus master follow the VMEbus specifications.

The bus request level is jumper-selectable, and the bus request daisy-chain is supported on all levels.

As bus master, the VME controller supplies 24-bits of address.

The VMEbus requester arbitrates on every transfer.

System Reset (SYSRESET*) clears any outstanding bus requests.

Bus Clear (BCLR*) and Bus Release (BREL*) are not supported.

4.2.11 Interrupter

The interrupter is compatible with the VMEbus specification.

The interrupt level is jumper-selectable.

The interrupt vector is dynamically programmable with any 8-bit vector. The onboard MPU sets the vector which causes the interrupt.

The interrupter supports the interrupt acknowledge daisy-chain.

4.2.12 Bus to Module Signaling Interrupt

Host access to the upper 512 bytes of the 128Kb memory window on the MVME330 causes an interrupt to the MPU on the MVME330.

- Access to the window does not constitute an MVME330 memory access.
- The host will not be forced to wait for data transfer acknowledge.
- Read will result in no data, write will not alter memory.
- Response is guaranteed to the host.
- An interrupt is generated to the MVME330 MPU.

4.2.13 Memory Map

The MVME330 subtends 128Kb of VME memory space.

The module 128K response address is jumper-selectable anywhere in the system bus address space.

As bus master, the MVME330 may access most of the bus address space (see Memory Map, Figure 5-3).

The onboard memory allocation is flexible; however, the MPU interrupt vectors must be in RAM. This necessitates a ROM/RAM switch after reset.

4.3 FIRMWARE

The resident firmware in the MVME330 is in two distinct sections, Diagnostic Self Test, and Kernel. Although not firmware, the Error Control Block is also described in this section.

4.3.1 Diagnostic Self Test

The presence of a parity error during power on tells the MVME330 to initiate the diagnostic self tests. The MVME330 executes a series of ROM-based tests to determine that the module is functioning properly. These tests include:

- LANCE register and loopback test.
- MPU test.
- Memory test for the 128K dynamic memory.
- EPROM checksum test.
- Control and Status register test.

If the diagnostic self test is successfully completed, the RUN LED will be lit and Kernel program execution will proceed.

If a test other than parity error fails, the diagnostic self test will stop, the Error Control Block will be written and Kernel program execution will proceed.

DRAM refresh interrupt performs a software read of addresses $400 to $600 approximately every 2 milliseconds. A parity error interrupt will be generated if a parity error is detected.

If a parity error interrupt occurs, Kernel or protocol software processing will continue, the Resultant Code in the Error Control Block will be written (refer to paragraph 4.3.2) and the RUN LED will extinguish and remain off until the MVME330 is reset or restarted.

4.3.1.1 Random Number DRAM Pattern Test

A pseudo random number pattern is written between memory locations $F01080 and the top of dynamic memory and then read back and compared with the original pattern.

Buffer comparison errors, error addresses and received values are reported (refer to paragraph 4.3.2).

4.3.1.2 Walking Bit DRAM Pattern Test

A single bit is rotated through a 16-bit word and written to memory locations between F01080 and the top of dynamic memory. The data is read and verified with the original pattern.

Any comparison error is reported. The address at which the comparison failure occurred, and the expected and received values are also reported (refer to paragraph 4.3.2).

4.3.1.3 DUART Internal Loopback

The SC68681 DUART on board is initialized to internal loopback mode and a pattern transmitted from channels a and b.

Each byte received is compared with the transmitted byte and any discrepancies reported (refer to paragraph 4.3.2).

4.3.1.4 Control and Status Register Test

The MVME330 Control and status registers which generate autovectored interrupts 1 through 4 are written. A counter is ticked when the corresponding interrupt occurs and compared with the expected value. Interrupt generation is repeated twice. The timer is then initialized on the SC68681 and the interrupt autovector 5 is verified. The Ethernet controller is then initialized and its ability to interrupt at autovector 6 is verified.

The error address and the type of error is reported (refer to paragraph 4.3.2).

4.3.1.5 LANCE Controller Test

This diagnostic performs a series of tests on the LANCE device in internal loopback mode. Transmit and receive buffer sizes are varied in order to test the buffer chaining capabilities of the device. The tests are:

1. Two receive buffers, 16 and 4 bytes long, are chained together for a 20-byte transmitted packet, including CRC.
2. Two 16 and 4 byte receive buffers are chained for a 24-byte transmitted packet. LANCE buffer error status is verified.
3. Four (16, 2, 2, and 4 byte) buffers are placed in the receive ring. The four byte buffer is assigned host ownership, thereby disallowing LANCE use of the buffer. A 24-byte packet is transmitted and the buffer error status is verified.
4. Three (16, 2 and 2 byte) receive buffers are chained to accept a 24-byte transmitted packet. Transmit and receive buffers are compared to validate data transmission.

Addresses used by the LANCE loopback test are:

. Initialization block - 600

. Receive Message Descriptor - 680

. Transmit Message Descriptor - 690

. Transmit Buffer - 700

. Receive Buffer - 800

The type of error is reported (refer to paragraph 4.3.2).

4.3.2 Error Control Block

Error status is stored in a control block accessible by the VMEbus host. The Resultant Code is set to zero when the diagnostic self tests are successfully completed.

To make sure the MVME330 is operating properly, the host processor may read the control block before downloading the protocol software and starting program execution in the MVME330. The error control block is read from the VMEbus at $1010 above the MVME330 slave address.

Table 4-1 describes the format of the Error Control Block, and Table 4-2 gives the interpretation of any errors reported by the diagnostic self test.

TABLE 4-1. Error Control Block Format

OFFSET	DESCRIPTION
0	Reserved (word)
2	Test Number (word)
4	Result Code (long word)
8	Reserved (long word)
12	Reserved (long word)
16	Error Address (long word)
20	Expected Result (word)
22	Received Result (word)

NOTE: OFFSET is the location above XXXX1010 where XXXX is the MVME330 slave address.

TABLE 4-2. Self Test Diagnostics Error Reporting

RESULT CODE	ERROR ADDRESS	EXPECTED RESULTS	RECEIVED RESULTS	REMARKS
1001	-	-	-	Parity error.
2001	(given)	(given)	(given)	Pattern Comparison Error. Random Number DRAM Pattern Test.
3002	(given)	(given)	(given)	Buffer Comparison Error. Walking Bit DRAM Pattern Test.
4001	-	(given)	(given)	Transmit and receive bytes comparison error. DUART Internal Loopback test.
5002	FDA	-	-	Level 1 interrupt incorrect. Control and Status Register test.
5003	FDC	-	-	Level 2 interrupt incorrect. Control and Status Register test.
5004	FDE	-	-	Level 3 interrupt incorrect. Control and Status Register test.
5005	FD0	-	-	Level 4 interrupt incorrect. Control and Status Register test.
5006	FE2	-	-	Improper timer count. Control and Status Register test.

TABLE 4-2. Self Test Diagnostics Error Reporting (cont'd)

RESULT CODE	ERROR ADDRESS	EXPECTED RESULTS	RECEIVED RESULTS	REMARKS
5007	FE4	-	-	Improper Ethernet controller interrupt count. Control and Status Register test.
6X07 X = test number	FFFFFFF3	-	-	Bus trap during LANCE transfer. LANCE Controller Test.
6X08 X = test number	-	-	-	Comparison error between received and transmitted packets. LANCE Controller Test.
FFFFFFF3	-	-	-	Bus trap during Ethernet controller access. Control and Status Register test.
FFFFFFF4	(given)	-	-	Bus trap during memory write. Random Number DRAM Pattern Test.
FFFFFFF4	(given)	-	-	Bus trap during memory access. Walking Bit DRAM Pattern Test.
FFFFFFF5	(given)	-	-	Bus trap during memory read. Random Number DRAM Pattern Test.
FFFFFFFC	-	-	-	Bus trap during transmit register access. Control and Status Register test.
FFFFFFFD	-	-	-	Bus trap during Receive register access. Control and Status Register test.
FFFFFFFE	SC68601 Address	-	-	Bus trap error. Control and Status Register test.
FFFFFFFF	(given)	-	-	Bus trap error. DUART Internal Loopback test.

4.3.3 Kernel Firmware

The MVME330 Kernel supplies functions which control and monitor the hardware features of the MVME330. The Kernel:

. Manages LANCE status registers and descriptor rings.

. Performs software refresh and timer functions.

. Manages interrupts.

. Allows protocol software to be written in a high-level language such as Pascal or C.

. Allows downloading over Ethernet or directly over the VMEbus from the host.

. Gathers running statistics on all conditions reported by LANCE.

CHAPTER 5 - SUPPORT INFORMATION

5.1 INTRODUCTION

This chapter provides connector signal descriptions and a parts location diagram for the MVME330.

5.2 CONNECTOR SIGNAL DESCRIPTIONS

This section provides information on the VME pin assignments and the Transceiver Cable pin assignment.

5.2.1 VME Pin Assignments

The MVME330 is inserted in the VMEbus P1 and P2 backplane sockets. The MVME330 connector P1 signals are identified in Table 5-1 by pin number, signal mnemonic, and signal name and description. Table 5-2 identifies connector P2 signals.

5.2.2 Transceiver Cable Pin Assignment

By specification, the MVME330 has a 15-pin female connector (MIL-C-24308 or commercial equivalent) with a slide latch assembly. Transceiver connector pin assignments are given in Table 5-3.

TABLE 5-1. Connector P1 VMEbus Pin Assignments

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
A1-8		D00-D07 DATA bus (bits 0-7) - Bidirectional data lines that provide data transfer to the VMEbus.
A9	GND	GROUND
A10		Not used.
A11	GND	GROUND
A12	DS1*	DATA STROBE 1 - input signal that indicates a data transfer on data bus lines D08-D15.
A13	DS0*	DATA STROBE 0 - input signal that indicates a data transfer on data bus lines D00-D07.

TABLE 5-1. Connector P1 VMEbus Pin Assignments (cont'd)

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
A14	WRITE*	WRITE - input signal that specifies the direction of data transfers.
A15	GND	GROUND
A16	DTACK*	DATA TRANSFER ACKNOWLEDGE - this output signal indicates that valid data is available on the data bus during a read cycle, or that data has been accepted from the data bus during a write cycle.
A17	GND	GROUND
A18	AS*	ADDRESS STROBE - the falling edge of this input signal indicates a valid address is present on the address bus.
A19	GND	GROUND
A20	IACK*	INTERRUPT ACKNOWLEDGE - input signal that indicates a VME interrupt acknowledge cycle. The VME system controller has been interrupted on one of seven levels and is now acknowledging the specific interrupt with a service routine.
A21	IACKIN*	INTERRUPT ACKNOWLEDGE IN - IACKIN* AND IACKOUT* form a daisy-chained acknowledge. The standard Motorola VMEbus backplane must have jumpers installed to continue the daisy-chained interrupt acknowledge beyond vacant card slots. This input signal is used when the controller module forms part of the daisy-chained interrupt acknowledge sequence.
A22	IACKOUT*	INTERRUPT ACKNOWLEDGE OUT - IACKIN* AND IACKOUT* form a daisy-chained acknowledge. The standard Motorola VMEbus backplane must have jumpers installed to continue the daisy-chained interrupt acknowledge beyond vacant card slots. This input signal is used when the controller module forms part of the daisy-chained interrupt acknowledge sequence.

TABLE 5-1. Connector P1 VMEbus Pin Assignments (cont'd)

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
A23	AM4	ADDRESS MODIFIER (bit 4) - one of the three-state input lines that provide additional information about the address bus such as size, cycle type, and/or data transfer bus master identification.
A24	A07	ADDRESS bus (bit 7) - one of 16 three-state input lines that specify an address in the memory map. Only the least significant 16 bits of the 23 associated VMEbus address lines are employed on the controller module.
A25	A06	ADDRESS bus (bit 6) - same as A07 on pin A24.
A26	A05	ADDRESS bus (bit 5) - same as A07 on pin A24.
A27	A04	ADDRESS bus (bit 4) - same as A07 on pin A24.
A28	A03	ADDRESS bus (bit 3) - one of 16 three-state input lines that specify an address in the memory map. During an interrupt acknowledge cycle, address bus lines A01-A03 are used to indicate the interrupt level that is being acknowledged.
A29	A02	ADDRESS bus (bit 2) - same as A03 on pin A28.
A30	A01	ADDRESS bus (bit 1) - same as A03 on pin A28.
A31	-12 VDC	-12 Vdc Power - used by LANCE devices.
A32	+5 VDC	+5 Vdc Power - used by the logic circuits on the controller module.
B1	BBSYS*	BUS BUSY - This signal is driven low when the controller module is the bus master. This signal is an input to the arbiter to indicate that the bus may be arbitrated.
B2		Not Used.
B3	ACFAIL*	AC FAILURE - Open collector driven signal which indicates that the AC input to the power supply is no longer being provided or that the required input voltage levels are not being met.

TABLE 5-1. Connector P1 VMEbus Pin Assignments (cont'd)

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
B4	BG0IN*	BUS GRANT 0 IN - Bus-grant-in and bus-grant-out form a daisy-chained bus grant. A grant received at the jumpered level indicates the MVME330 may become the bus master. The remaining three bus-grant-in lines are connected directly to their respective bus-grant-out lines.
B5	BG0OUT*	BUS GRANT 0 OUT - Bus-grant-in and bus-grant-out form a daisy-chained bus grant. When a bus-grant-in is received at the jumpered level and the MPU is not waiting for bus mastership, the bus-grant-out signal is true on the respective level.
B6	BG1IN*	BUS GRANT 1 IN - same as BG0IN on pin B4.
B7	BG1OUT*	BUS GRANT 1 OUT - same as BG0OUT on pin B5.
B8	BG2IN*	BUS GRANT 2 IN - same as BG0IN on pin B4.
B9	BG2OUT*	BUS GRANT 2 OUT - same as BG0OUT on pin B5.
B10	BG3IN*	BUS GRANT 3 IN - same as BG0IN on pin B4.
B11	BG3OUT*	BUS GRANT 3 OUT - same as BG0OUT on pin B5.
B12-B15	BR0*-BR3*	BUS REQUEST (0-3) - the bus request at the jumpered level is true when the MPU requires bus mastership. When one or more bus request lines is true in the ROR mode, bus mastership is released. When the controller module is the system controller, bus request level three is monitored by the arbiter.
B16-B19	AM0-AM3	ADDRESS MODIFIER (bits 0-3) - same as AM4 on pin A23.
B20	GND	GROUND
B21,B22		Not Used.
B23	GND	GROUND

TABLE 5-1. Connector P1 VMEbus Pin Assignments (cont'd)

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
B24-B30	IRQ7*-IRQ1*	INTERRUPT REQUEST (7-1) - seven prioritized interrupt request inputs. Jumper enabled, level seven is the highest priority.
B31		Not used.
B32	+5 VDC	+5 VDC Power - same as +5 VDC on pin A32.
C1-C8	D08-D015	DATA bus (bits 8-15) - eight of 16 three-state bidirectional data lines that provide the data path between VMEbus master and slave.
C9	GND	GROUND
C10	SYSFAIL*	SYSTEM FAIL - reflects state of FAIL bit in MCR and fail indicator. When enabled in MCR, this bidirectional signal generates an interrupt request.
C11	BERR*	BUS ERROR - an active low output signal that indicates an error has occurred during a data transfer cycle.
C12	SYSRESET*	SYSTEM RESET - the system controller provides this input signal that causes a board level reset on the controller module.
C13	LWORD*	LONGWORD - three-state driven signal to indicate that the current transfer is a 32-bit transfer.
C14	AM5	ADDRESS MODIFIER (bit 5) - same as AM4 on pin A23.
C15	A23	ADDRESS bus (bit 23) - same as A07 on pin A24.
C16	A22	ADDRESS bus (bit 22) - same as A07 on pin A24.
C17	A21	ADDRESS bus (bit 21) - same as A07 on pin A24.
C18	A20	ADDRESS bus (bit 20) - same as A07 on pin A24.
C19	A19	ADDRESS bus (bit 19) - same as A07 on pin A24.
C20	A18	ADDRESS bus (bit 18) - same as A07 on pin A24.

TABLE 5-1. Connector P1 VMEbus Pin Assignments (cont'd)

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
C21	A17	ADDRESS bus (bit 17) - same as A07 on pin A24.
C22	A16	ADDRESS bus (bit 16) - same as A07 on pin A24.
C23	A15	ADDRESS bus (bit 15) - same as A07 on pin A24.
C24	A14	ADDRESS bus (bit 14) - same as A07 on pin A24.
C25	A13	ADDRESS bus (bit 13) - same as A07 on pin A24.
C26	A12	ADDRESS bus (bit 12) - same as A07 on pin A24.
C27	A11	ADDRESS bus (bit 11) - same as A07 on pin A24.
C28	A10	ADDRESS bus (bit 10) - same as A07 on pin A24.
C29	A09	ADDRESS bus (bit 09) - same as A07 on pin A24.
C30	A08	ADDRESS bus (bit 08) - same as A07 on pin A24.
C31	+12 VDC	+12 Vdc Power - used by the logic circuits on the controller module.
C32	+5 VDC	+5 Vdc Power - same as +5 Vdc on pin A32.

NOTES: 1. All signal directions are with respect to the MVME330.

2 The bus master is the device on the VMEbus that is in control of the transaction at the present time.

TABLE 5-2. Connector P2 VMEbus Pin Assignments

PIN NUMBER	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
A1-A32		Not Used
B1	+5 VDC	+5 Vdc Power - used by the MVME330.
B2	GND	GROUND
B3-B11		Not Used
B12	GND	GROUND
B13	+5 VDC	+5 Vdc Power - same as +5 Vdc on pin B1.
B14-B21		Not Used
B22	GND	GROUND
B23-B30		Not Used
B31	GND	GROUND
B32	+5 VDC	+5 Vdc Power - same as +5 Vdc on pin B1.
C1	XRES	EXTERNAL RESET - causes the MPU to be reset.
C2	ABOP	This signal, when low, generates a level 7 interrupt, causing the software to abort.
C3-C32		Not Used

NOTE: "Not used" denotes signal pins not applicable to the MVME330.

TABLE 5-3. Transceiver Connector Pin Assignments

PIN NUMBER	SIGNAL NAME
1	Shield (ground) (NOTE)
2	Collision Presence
3	Transmit +
4	Ground
5	Receive +
6	Power return (ground)
7	Reserved
8	Ground
9	Collision Presence -
10	Transmit -
11	Reserved
12	Receive -
13	Power (+12 Vdc)
14	Ground
15	Reserved

NOTE: Shield must be terminated to connector shell as well as pin 1 on both ends, and must not be connected to any other pin. No connections are allowed on the reserved pins.

5.3 PARTS FOR THE MVME330

Components for the MVME330 are shown in Figure 5-1, Parts Location Diagram. Figure 5-1 reflects the latest issue of hardware at time of printing.

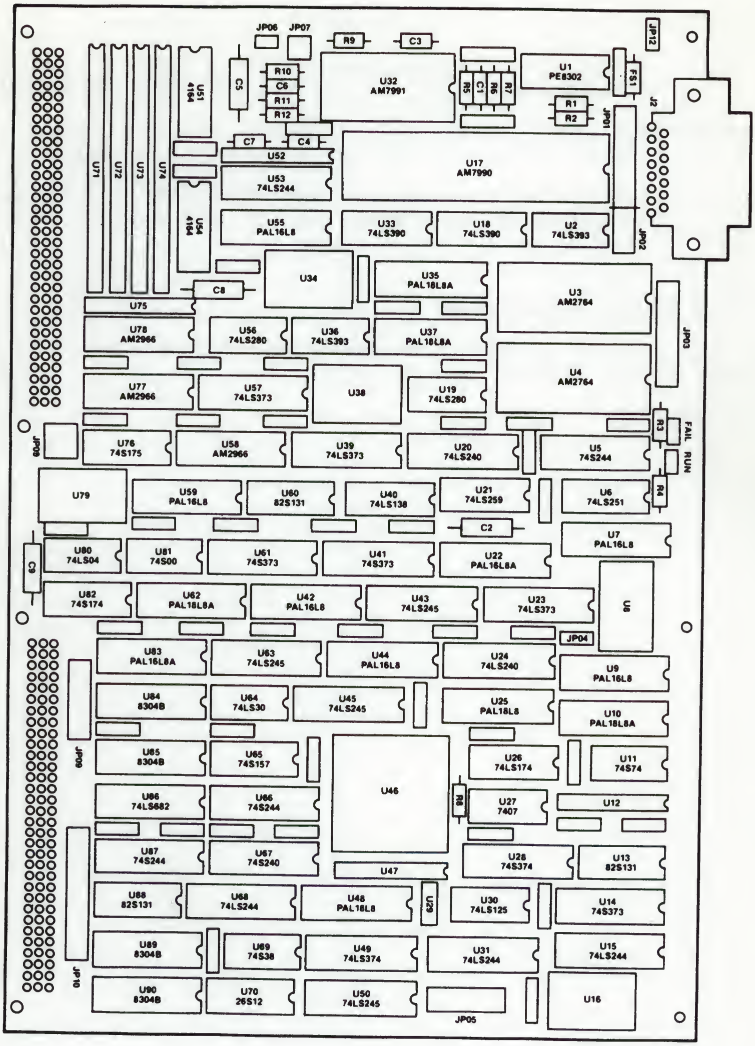


FIGURE 5-1. Parts Location Diagram

5.4 ASSOCIATED CABLES FOR THE MVME330

Associated cables and connectors for the MVME330 are as follows.

1	FEMALE	15 conductor "D" subminiature type (MIL-C-24308 or equivalent) conductor with a slide latch assembly mounted on the MVME330.
1	standard	transceiver cable: Four stranded, shielded, twisted pair conductors plus an overall shield and insulating jacket.
1	MALE	15 conductor "D" subminiature type (MIL-C-24308 or equivalent) conductor with locking posts, part of the transceiver cable.

5.5 OUTLINE DRAWING AND MEMORY MAP

Figure 5-2 shows the overall outline dimensions for the MVME330. The system memory map is shown in Figure 5-3.

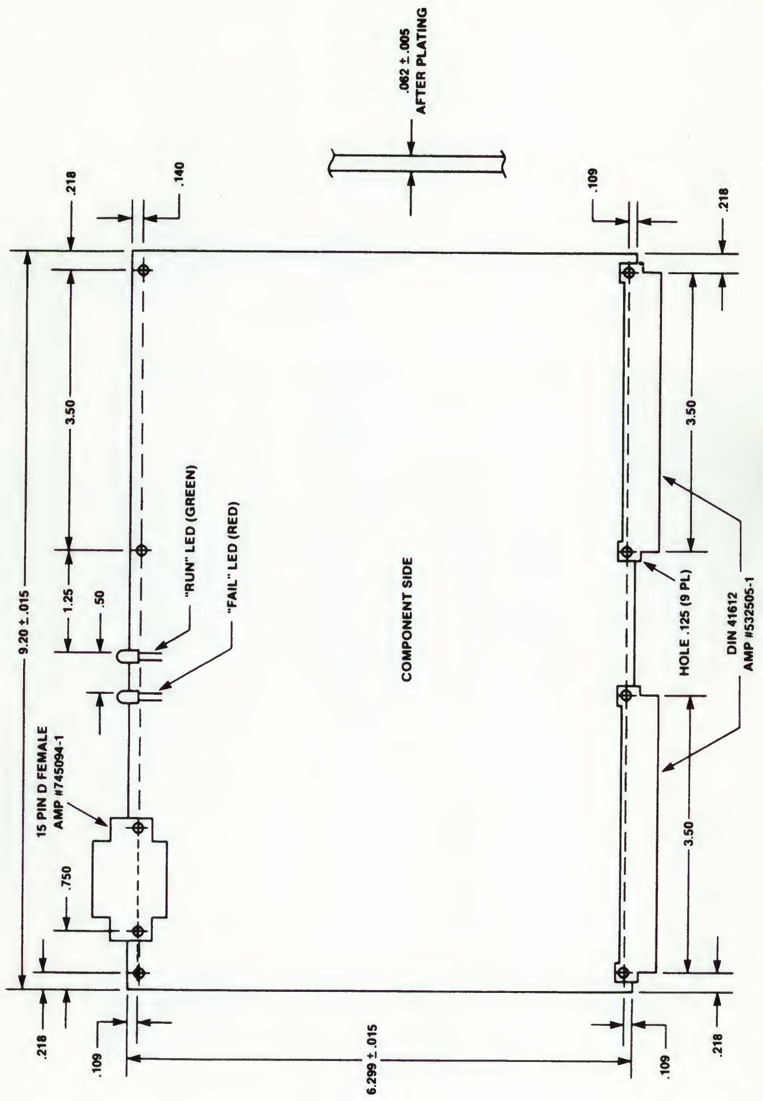
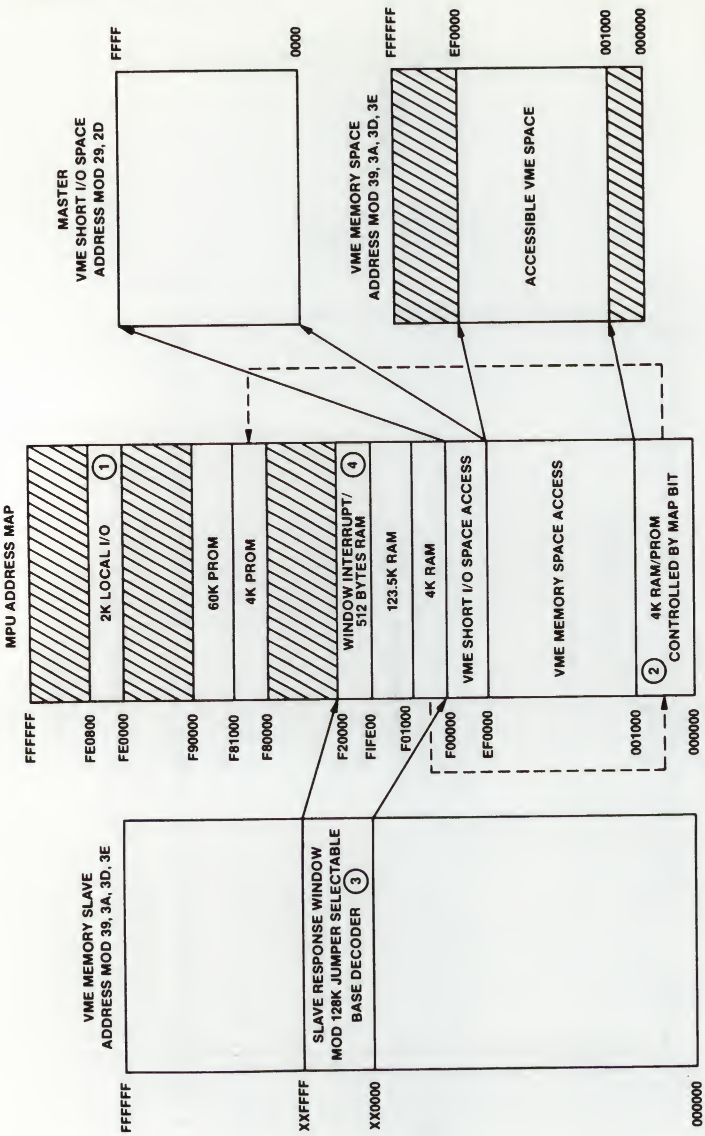


FIGURE 5-2. MVME330 Outline Drawing



NOTES:

1. LOCAL I/O SPACE 2MS TIMER, LANCE I/O REGISTERS, MPU CONTROL REGISTERS, ETC.
2. MAP BIT BRINGS 4K OF RAM @ F00000 DOWN TO 000000 AND MOVES 4K OF PROM TO F80000. RESET ON POWER UP.
3. MPU CANNOT ACCESS ITS OWN SLAVE RESPONSE WINDOW IN VME ADDRESS SPACE.
4. VME SLAVE ACCESS TO THIS 512 BYTE SEGMENT CAUSES AN AUTO VECTOR INTERRUPT TO MPU WITHOUT ACCESSING THE INTERNAL MPU BUS. THIS SEGMENT IS THE TOP 512 BYTES OF RAM FOR THE MPU.

FIGURE 5-3. MPU Memory Map

APPENDIX A - BUS INTERFACE PROTOCOL (BIP)

A.1 INTRODUCTION

This appendix defines the protocol used by a processor, or host, to communicate with an MVME330 in order to manage network communications. It does not define the functions performed on either side of the interface before or after receiving the data. It defines a method of passing data within the context of a communication protocol and is the definitive document for writing an I/O driver for the MVME330.

The MVME330 performs several levels of telecommunication protocol processing to offload the burden from the host. The Bus Interface Protocol (BIP) is a host-to-MVME330 software interface which provides an efficient mechanism for controlling the offloaded protocol, without using any protocol-specific features. In this way, a protocol running on the MVME330 may be changed while the operating system I/O calls remain the same. The user specifies the protocol and operations using system I/O calls, and the host I/O driver forwards these requests to the MVME330.

The first protocol planned for the MVME330 is the Xerox Network System's Sequenced Packet Protocol, including all lower layers. Future protocols to be implemented using the software interface defined herein include ECMA-72 Transport layer and ARPAnet's TCP/IP.

A.1.1 General Description of Communication Protocols and Usage

The Open Systems Interconnection (OSI) seven layer model for communications protocols provides a useful frame of reference for a discussion of protocols. Briefly, starting from the lowest layer, the seven layers are: the physical layer, in this case Ethernet 2.0; the data link layer, defining the control of the transmission and the lowest level addressing formats; the network layer, which provides for end-to-end addressing and multiple network access; the transport layer, which provides reliable sequenced message delivery; the session layer, defining the steps used during a communication session; the presentation layer, which translates, if necessary, data formats so that dissimilar devices can communicate with each other; and finally the application layer, which is basically the function which the user wishes to perform using the lower levels, such as file transfer.

Since the session layer deals closely with the operating system in the sense that it must cause tasks to execute upon request, while the transport layer deals more with the actual transmission and reception of data, this is a good breaking point between those functions executed in the host and those functions performed by the front-end processor. The transport layer performs the functions of reliable transmission and message acknowledgement, and when a transport layer message is not acknowledged within some protocol-specific interval, the message is retransmitted. Thus, not only must outgoing messages be passed down to the network layer for routing, but they must also be saved for possible retransmission until they are acknowledged by the destination or transport layer. Given that the MVME330 has a timer function and the ability to read and write host memory, it is an ideal device to implement the transport layer, as well as the lower levels for Ethernet.

Traditionally, the "I/O Driver" has been the code which deals directly with a peripheral I/O device. More specifically, it has been the code which interfaces an I/O device to an operating system, presenting commands and data to a peripheral I/O device. This is the case with the MVME330 as well; however, the MVME330 performs operations and possibly transformations on the data, such as appending protocol-specific data headers to outgoing messages before they appear on the media, which in this case is Ethernet. The important point is that the I/O driver doesn't know anything about the content of the data; it is merely concerned with the exchange of the data between the user task in the host and the peripheral device. If the I/O driver can be defined such that the particular protocol in use need not be known, a separation of function ensues, allowing modular software development. This is the case with the MVME330. When using this document to develop protocols, it is recommended that the developer bear in mind this separation of functions.

A.2 ENVIRONMENT AND INTERFACES

The MVME330 software interface enables host processor to control Ethernet communications and protocol processing in an MVME330. The host is the controlling device since it initiates all activity and determines when data transfers take place between itself and the MVME330. Most common operating systems which are capable of handling asynchronous communications should be able to support an MVME330 using this software interface protocol.

Since the MVME330 contains an MC68000 or MC68010 MPU, it is capable of more intelligent activity than simply acting as a front-end protocol Microprocessor Unit (MPU). For example, two MVME330's can be configured on a common system bus as a protocol gateway from one Ethernet to another. Nothing in the original design precludes such a configuration; indeed, this is a configuration planned as an area for future development. Other types of gateways can be imagined also, which may call for the MVME330 to act as an intelligent bus master, controlling various sorts of devices and device controllers on a system bus.

A.2.1 Interface Characteristics

The MVME330 software interface can be characterized as simple yet flexible. The number and location of the I/O mailboxes is negotiated at start up time, allowing the host to specify its preference for their location. Once the I/O mailbox negotiation is complete, the I/O mailboxes are used to exchange data during a transaction. The MVME330 or the host will move whatever data is described through the use of the I/O mailbox. This allows the user to specify the protocol family (such as XNS) and level (e.g., sequenced packet) desired, thus allowing the I/O driver to be protocol independent while allowing the user access to all protocol levels present on the MVME330. Additionally, each side of the interface has the ability to request the other side to interrupt or not, upon completion of the transaction, so interrupt overhead can be finely tuned for a specific host-MVME330 combination.

As a result of the mailbox negotiation phase, two sets of I/O mailboxes are defined: one set for host-to-MVME330 requests and one set for MVME330-to-host requests. This provides a symmetric implementation, allowing for multiple MVME330 and other configurations. This fact, along with the level of indirectness provided by placing addresses of data structures in the mailboxes, provides maximum flexibility, configurability, and expandability.

A.3 FUNCTIONAL SPECIFICATION

The software protocol described in this document binds together a host device driver and a communication protocol program running in an MVME330.

A.3.1 Software Configurations and Modes of Operation

The MVME330 Kernel resides in ROM on the MVME330 module. Application programs, including the IP I/O executive and the protocol application program, may be downloaded into MVME330 memory and started in cooperation with the Kernel. Alternatively, they too may reside in ROM.

Three modes of operation allow for the downloading of an application program, the negotiation of communication parameters, and the exchange of Ethernet packets.

The MVME330 software operates in three major modes:

a. Initialization mode
b. Negotiation mode
c. Operation mode

A.3.1.1 Initialization Mode

When the MVME330 is reset, for instance upon first application of power, the ROM-based Kernel prepares the MVME330 to receive the application protocol software that will reside in MVME330 RAM. Once the Kernel has given permission, the application protocols will enter the MVME330 RAM in one of these ways:

. Downloaded from the host
. Uploaded from the network
. Transferred from the ROM

In all three cases, the application software will be moved to MVME330 RAM, and it will issue a command to the MVME330 Kernel to start the application running. (Permission to load is not granted if an application has already been loaded.)

A.3.1.2 Negotiation Mode

When the application code is first started, usually upon being loaded into the MVME330, it prepares to negotiate with the host so as to determine the characteristics desired for operation mode. The host uses a mailbox that is defined by the MVME330 for use during negotiations, and initializes all values in the mailbox, as well as modifying two addresses in the firmware. The application program examines these definitions and responds by assigning a unique host number and filling in any default values. One of the negotiated items is the location (and number) of other mailboxes to be used in the operation mode.

A.3.1.3 Operation Mode

When both the host and the MVME330 have completed negotiations, normal operations take place by means of the newly negotiated mailboxes. Requests to open, read, and write can be made to the MVME330 application program.

A.4 BIP OPERATIONS AND COMMANDS

A.4.1 Initialization

The MVME330 provides three different methods for initializing the applications program code in the MVME330 memory:

a. Downloading Mode
b. Uploading Mode
c. Resident Mode

A.4.1.1 Downloading Mode

When the MVME330 is reset, for instance upon first application of power, the ROM-based Kernel gains control and prepares the device for downloading from a host processor. Using the program "ENPLD" (for MVME330 LOAD), the host acquires permission to download from the MVME330 Kernel, loads the application program code into the MVME330 memory, and instructs the Kernel to start the application running. (Permission to download is not granted if an application has already been loaded.)

A.4.1.2 Uploading Mode

Not available this release.

A.4.1.3 Resident Mode

Application programs in ROM begin running upon reset.

A.4.1.4 Initialization Procedure

In order to begin communication, the MVME330 and the host must agree on a series of address locations that provide a common ground for data sharing. There are five physical bus addresses, residing in the MVME330, which have been arbitrarily assigned. They are described using the format "*XXXXnnnn*" which indicates that they are offsets from the starting address of the MVME330. *XXXX* is defined individually for each operating system and bus structure. The following are the five locations and their values. Figure A-1 illustrates the initialization address blocks.

a.	Kernel JUMP command location	*XXXX*1000
b.	Firmware start address location	*XXXX*1004
c.	Default negotiation mailbox location	*XXXX*1008
d.	Default host mailbox location	*XXXX*1018
e.	Default MVME330 mailbox location	*XXXX*1038

```
|<----------32---------->|
|-----|------|-----|-----|
|                        |  Kernel JUMP command
|                        |  location
|-----|------|-----|-----|
|                        |  Firmware start address
|                        |  location
|-----|------|-----|-----|
|                        |
|                        |  Default negotiation mailbox
|                        |
|-----|------|-----|-----|
|                        |
|                        |  Default host mailbox
|                        |
|-----|------|-----|-----|
```

FIGURE A-1. Initialization Address Blocks

a. Kernel JUMP Command Location XXXX1000

This address defines the location that the MVME330 Kernel is continually monitoring whenever the MVME330 is reset. The host will load the command "JUMP" into this location when it is time for the MVME330 Kernel to begin its role in the initialization.

This value is originally set to 0 by ENPLD. The host will change the value to 80800000 to begin the execution of the firmware during the negotiation procedure. After the firmware has been started, the MVME330 Kernel will reset this location as necessary for operation.

b. Firmware Start Address Location XXXX1004

The MVME330 RAM address defines the location to which the MVME330 Kernel is to "JUMP" when it begins its role in the initialization.

This value is originally set to the internal MVME330 address of the main entry point of the protocol firmware by ENPLD. After initialization, this address is used as a Kernel/firmware communication pointer.

c. Default Negotiation Mailbox Location XXXX1008

This address defines the location that the host and the MVME330 will use for the negotiation communication. Each will place information into and remove information from a mailbox at this point.

During initialization, the physical bus address is set by the device driver, based upon values that were established during the original system configuration.

d. Default MVME330 Mailbox Location XXXX1038

This address defines the location that will be used for all data transfers and communications from the host to the MVME330, if no other address is specified by the host.

During initialization, this value is set by the MVME330 based upon values that were established during the original system configuration.

A.4.2 Negotiation

The negotiation procedure establishes a firm list of mailboxes for use by the MVME330 and the host. These values are transferred from one to another via the negotiation mailbox. Figure A-2 illustrates the negotiation mailbox.

```
|<----16---->|<----16---->|
|------|-----|
|            |     Version
|------|-----|
|            |     Host Id
|------|-----|
|            |     Host Mailbox Count
|------|-----|
|            |     MVME330 Mailbox Count
|------|-----|------------|
|                         |  Host Mailbox Address
|------|-----|------|-----|
|                         |  MVME330 Mailbox Address
|------|-----|------|-----|
```

FIGURE A-2. Negotiation Mailbox

A.4.2.1 Version Number

This value defines a 16-bit number that has been set by the host for a two-fold purpose. The version numbers are unique for each release and are stored in the header files. The version number is compared to the value in the OS Kernel to determine if this release of the protocol software can operate on this operating system. First, it is used (by a test-and-set protocol) by each host to arbitrate access to the negotiation mailbox. Secondly, it allows the MVME330 to determine the byte-swap nature of the host.

A.4.2.2 Host ID

This value defines a 16-bit number that is set by the MVME330 during the negotiation procedure to uniquely identify each host on the network. The value placed in the mailbox by the MVME330 will be used to identify all ensuing I/O requests by the host.

A.4.2.3 Host Mailbox Count

This value defines a 16-bit number that establishes the number of host mailboxes that the host desires to use. If this count is set equal to zero, the default value of one mailbox is used.

A.4.2.4 MVME330 Mailbox Count

This value defines a 16-bit number that establishes the number of MVME330 mailboxes that the host desires to use. If this count is set equal to zero, the default value of one mailbox is used.

A.4.2.5 Host Mailbox Address

This value is a 32-bit physical bus address that is set by the host to define the starting location of the host mailboxes. If this address is set equal to zero, the default value that the MVME330 specified in the initialization addresses is used.

When multiple host mailboxes are established, their addresses are assumed to be contiguous, starting with the location specified in this field.

A.4.2.6 MVME330 Mailbox Address

This value is a 32-bit physical bus address that is set by the host to define the starting location of the MVME330 mailboxes. If this address is set equal to zero, the default value that the MVME330 specified in the initialization addresses is used. When multiple mailboxes are established, their addresses are assumed to be contiguous, starting with the location specified in this field.

A

A.4.2.7 Negotiation Procedure

In the negotiation mailbox, the host sets the version number and the host ID to zero, the host and MVME330 mailbox counts to the appropriate values, the host and MVME330 mailbox addresses to the corresponding locations and the firmware start address to the firmware entry point address. The host then sets the Kernel JUMP command to 80800000.

The MVME330 Kernel recognizes this new command and begins the initialization process. The Kernel will read the firmware start address, reset the command in the JUMP field, signaling the host that it has received the command, load the address for the starting subroutine that will be used to initialize the LANCE and build the reference tables that link the Kernel command to the LANCE in the second field, and then jump to the host specified address to begin execution. When the MVME330 has processed all of the values, it places the host ID in field two of the negotiation mailbox. The host, sensing the change in the ID, completes the negotiation procedure by transferring and starting the host ID for identifying future transactions.

A.4.3 Data Transfer

The data transfer procedure is the most used BIP operation. It facilitates the movement of data streams from the host to the MVME330 RAM and vice versa. It performs these transfers through mailboxes that were established during the negotiation procedure. Figure A-3 illustrates the data transfer mailbox.

```
|<----16---->|<----16---->|
|------|-----|
|            |    Command
|------|-----|
|            |    Host ID
|------|-----|------------|
|                         |  Status
|------|-----|------|-----|
|                         |  Socket ID
|------|-----|------|-----|
|                         |  Transaction ID
|------|-----|------|-----|
|                         |  Misc. Parameter/Result
|------|-----|------|-----|
|                         |  Source Address
|------|-----|------|-----|
|                         |  Destination Address
|------|-----|------|-----|
|                         |  Byte Count
|------|-----|------|-----|
```

FIGURE A-3. Data Transfer Mailbox

A.4.3.1 Command

This value is a 16-bit command code that is loaded alternately by the host and the MVME330. Table A-1 lists the commands that are used by the host to instruct the MVME330. Table A-2 lists the commands that are used by the MVME330 to instruct the host. The value is placed in the command field to initiate the action.

TABLE A-1. Host to MVME330 Command

COMMAND	VALUE
IDLE	0
OPEN	1
CLOSE	2
READ	3
WRITE	4
IOCTL	5
NOTIFY	6
OK	7
YOURS	8
ABORT	9

TABLE A-2. MVME330 to Host Commands

COMMAND	VALUE
IDLE	004
PENDING	814
IN	824
OUT	834
WAKEUP	864

By alternately using these commands, the MVME330 and the host can communicate and transfer data. The interaction of the commands is illustrated in the mailbox state transition diagram, Figure A-4.

```
                                            H           E
                                          ,-->(YOURS)------>+
                                         /       |          |
                                        /        |  E       |
          H                            /         V          |
(IDLE)------->(COMMAND)------->(IN/OUT)<---------+          |
  ^            /  |               |    \         ^          |
  |           /   |E              |H    \        | E        |
  |      E   /    V               V      \   H   |     E    V
  +<--------/  (ERROR)          (ABORT)   \--->(OK)-------->+
  ^               |               |                         |
  |               |H              |E                        |
  |               V               V                         V
  +<--------------+<--------------+<------------------------+
```

FIGURE A-4. Mailbox State Transition Diagram

a. IDLE (0 or 00H)

IDLE is the initial and final state. The mailbox state is set to IDLE when the mailbox is created.

b. Action Commands

1. OPEN

OPEN is issued by the host to establish a connection on the network. No data is transferred. Returns a DONE.

2. CLOSE

CLOSE is issued by the host to terminate a connection on the network. No data is transferred. Returns a DONE or a PENDING.

3. READ

READ is issued by the host to notify the MVME330 that it is ready to receive data from the MVME330.

4. WRITE

WRITE is issued by the host to notify the MVME330 that it is ready to send data to the MVME330.

5. IOCTL

IOCTL is issued by the host to notify the MVME330 that network control information must be exchanged.

6. NOTIFY

NOTIFY is issued by the host when it receives a pending command from the MVME330. NOTIFY requests the MVME330 to interrupt the host upon the completion of a particular address conversion that will occur in the future.

7. OK

OK is issued by the host when it has performed a data transfer from the MVME330. The host will also return the actual byte count of the data so the MVME330 can determine whether another transfer is needed to move the entire data.

8. YOURS

YOURS is issued by the host to notify the MVME330 to perform the data transfer on its own. Before issuing the command, the host also translates the addresses of the data to physical bus addresses. This command is only used on those systems where the MVME330 is capable of performing transfers without host assistance.

9. ABORT

ABORT is issued by the host when the MVME330 has requested impossible IN or OUT address. The MVME330 will return the mailbox to an idle state when it receives this command.

10. PENDING

PENDING is issued by the MVME330 when the host has requested a READ or WRITE command and the MVME330 cannot immediately return an IN or OUT answer. Upon receiving a PENDING, the host may issue a NOTIFY or an ABORT.

11. IN

IN is issued by the MVME330 in response to the host request for a WRITE. The IN signifies that the MVME330 has provided an MVME330 source address for the data transfer.

12. WAKEUP

WAKEUP is issued by the MVME330 in response to a previously issued PENDING/NOTIFY sequence. The WAKEUP signifies that the address conversion process is complete and the host may continue its role in the transfer.

A.4.3.2 Host ID

This value is the 16-bit number that was set by the MVME330 during the negotiation procedure. It is used by the host to identify all I/O requests.

A.4.3.3 Status

This value is a 32-bit completion status that is set by the MVME330. If no error occurs, a file descriptor or data length is returned. The error numbers are those assigned by XNS to signal XNS software. The action on the error codes are taken by the XNS software on the host.

A.4.3.4 Socket ID

This value is a 32-bit socket identifier that is assigned by the MVME330 in response to an OPEN command by the host. It is used by the host on every transfer to identify the socket that is being used.

A.4.3.5 Transaction ID

This value is a 32-bit quantity that is supplied by the host to identify each transaction. It is only used by the host.

A.4.3.6 Miscellaneous Parameter

This value is a 32-bit quantity that is used by the OK, IN or OUT command to return arguments.

A.4.3.7 Source Address

This value is a 32-bit address that is supplied alternately by the host and the MVME330 to identify the location of data to be moved in a transfer.

A.4.3.8 Destination Address

This value is a 32-bit address that is supplied alternately by the host and the MVME330 to identify the location to which data will be moved in a transfer.

A.4.3.9 Byte Count

This value is a 32-bit transfer byte count that is supplied alternately by the host and MVME330 to define the length of the data to be transferred.

A.4.3.10 **Data Transfer Procedure**

The process for communicating a command from the host to the MVME330, using the MVME330 mailboxes that were established during the negotiation procedure, is a series of simple steps.

a. Wait for mailbox command to become IDLE.

b. Set the HOST ID.

c. Set the socket ID as returned from a previous OPEN, or zero if this an OPEN.

d. Set the transaction ID to any value to be returned as an identifier of this transaction.

e. Set the miscellaneous parameter as required by the particular command.

f. Set the source or destination address and byte count for the user buffer.

g. Set command

h. Interrupt the MVME330.

i. Wait for the command code to change.

   1. If the command becomes IN, the MVME330 has provided an MVME330 destination address. Either perform the copy and set the command code to IDLE, or convert the host source address to a bus physical address and set the command code to YOURS and wait for the command to become IDLE.

   2. If the command becomes OUT, the MVME330 has provided an MVME330 source address. Either perform the copy and set the command code to IDLE, or convert the host destination address to a bus physical address and set the command code to YOURS and wait for the command to become IDLE.

   3. If the command becomes PENDING, set the command to NOTIFY and the MVME330 will issue an interrupt upon later completion. Wait for the command to become IDLE. Alternatively, set the command to ABORT and return the status to the user or automatically schedule a re-entry later.

   4. If the command becomes IDLE, return the status to the user.

      The process for communicating a command from the MVME330 to the host using the host mailboxes that were established during the negotiation procedure is also a series of simple steps.

j. If the command code is WAKEUP, use the mailbox contents to complete a previously issued PENDING/NOTIFY request.

The process for communicating a command from the MVME330 to the host, using the host mailbox(es) established during the negotiation procedure, is also a series of simple steps. See Figure A-5.

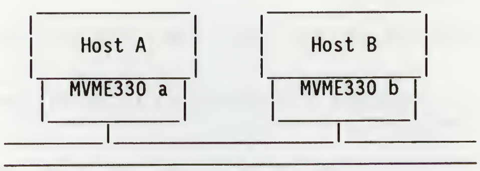


FIGURE A-5. Host Data Transfer

1. Host A wants to transfer data to host B. Host A(A) notifies the MVME330 a(a) to open a socket.
2. A returns the socket number that was opened.
3. A issues an OPEN command on that socket.
4. Host B(B) continually has an outstanding READ on its courier socket (see NOTE).
5. MVME330 b(b) receives the OPEN command on the courier (well-known) socket.
6. b assigns a new socket for communication.
7. b notifies B with a WAKEUP command.
8. B places a READ to the new communication socket.

NOTE

If no outstanding READ exists and the socket is not asynchronous, B will occasionally sample incoming sockets looking for incoming packets (known as polling). If no outstanding READ exists and the socket is asynchronous, b will notify B by interrupting with a "SIGNAL" command.

A.4.4 Run-time Procedures

The first commands executed by FUSION must be xbs-open, xdg-open, xms-flip, xpc-open and xsp-open. These commands define the socket connections, when using the run-time library functions.

A.4.5 Restrictions on Use

a. Negotiations must be satisfactorily completed prior to issuing I/O requests.

b. The Host device driver must wait for the completion status upon a CLOSE command to insure that all I/O on the system bus is complete.

A.4.6 Error Handling and Recovery

Error conditions can be easily dealt with, with due respect given to the restrictions above. Synchronous errors may result from invalid socket ID's, asynchronous errors may result from network problems.

A.4.7 Host/BIP Interaction

A.4.7.1 Initialization Addresses

Paragraph A.4.1 describes the initialization addresses that are used to define the negotiation mailboxes, as well as Begin System Initialization. The value of these addresses is a function of where the host operating system expects the bus memory to be located. The addresses are initialized through the use of jumpers on the MVME330. For a discussion of how the jumpers are to be set, refer to Chapter 2.

A.4.7.2 Address Usage

The physical and logical addresses are used by BIP to perform data transfers from the host to the MVME330 and vice versa. BIP works with two different addresses:

a. Physical bus address

b. MVME330 RAM Memory Access.

See Figure A-6. Data resides at point X in the MVME330 RAM. The MVME330 MPU accesses the data at X by using address B. The host accesses the data at X by using address A. Although both address A (the physical bus address) and address B (the local address) point to the same memory location, they are different values.

Each address is 32 bits. The lower 17 bits are the same for the local address and the bus address. It is the higher 15 bits that vary from system to system. These 15 bits are what the user sets with jumpers at configuration time, in accordance with how the host operating system has configured the bus. At execution time, it is the BIP responsibility to translate one address to another.

Although room for a 32-bit address has been provided, most systems only use a 24-bit address. For these systems, the lower 17 bits are the same and the higher 7 bits are system dependent.

A.5 SYSTEM DEPENDENT CONSTRAINTS

For the VERSAdos system, the upper level bits are set to 00DE.

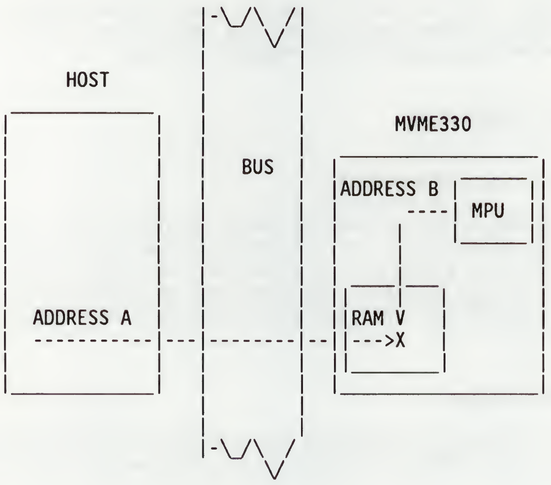


FIGURE A-6. Host/MVME330 Relationship

APPENDIX B - PROGRAMMING GUIDE

B.1 INTRODUCTION

If the MVME330 module is going to be used for a non-standard implementation, the MVME330 can be programmed through 11 specific control/status registers. If it is a standard implementation, the user can ignore this section.

The MVME330 local I/O space contains the control and status registers for MVME330 specific resources as well as access to the LANCE registers and the network parameters PROMs.

B.2 MVME330 CONTROL/STATUS REGISTERS

The MVME330 has 11 control/status registers. Details for usage and purpose of these registers appear in the following text.

B.2.1 FE0081: Vector Register (VECTR - 8 Bits)

This register, when written with the desired VME interrupt vector, causes an interrupt to the host at that vector. When read, this register returns the most significant 8-bits of the selected slave response base address (A23-A16). This value enables MVME330 operational software to know what its slave response address is, and thus discriminate between internal and external addresses. This register is also accessible at odd addresses from FE0083 to FE009F.

B.2.2 FE00A1: Control Status Register (CSR - 8 Bits)

The CSR contains four system control and status indicators. It is also accessible at odd locations from FE00A3 to FE00BF. Bit definitions for the CSR are given in Table B-1.

TABLE B-1. Control Status Register Bit Definitions

BIT	RW	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
7	R	VIRQ	Vectored Interrupt Request. This bit is set when the VECTR is written and cleared when the interrupt transaction on the VMEbus is complete. Another VME interrupt transaction cannot take place until this bit is cleared.
6	RW	FAIL	FAIL. This bit is set when the MVME330 is reset. It is also programmable to the reset state and drives SYSFAIL* on the VMEbus. When FAIL is on, the red LED (FAIL) is on and the green LED (RUN) is off. When FAIL is off, the red LED is off and the green LED is on.
5	RW	RESET	System RESET. This bit enables the MVME330 to reset the host and all other VMEbus resources. When a "1", this bit causes SYSRESET* to be asserted to all resources including the MVME330. Because the SYSRESET* pulse is less than VME minimum requirements, the use of this facility is not recommended.
4	R	TIMER	TIMER. Active low(0), this bit indicates when the 2 millisecond timer has gone off. It is cleared when EXR is written. The TIMER and all exceptions cause a level 7 autovector interrupt. The EXR must be written (clearing the TIMER bit) prior to executing an RTE.
3	RC	SUSPEND	SUSPEND. Active low (0), this bit is set each time the LANCE does DMA. It can be used to control software access to the host bus and is cleared with a 0 or reset.
2-0	Unused	Undefined.	

B.2.3 FE00C1: Interrupt Enable Register (IER - 1 Bit)

This is a single bit (bit 7) register used to enable all interrupts to the MPU. It is cleared by an MVME330 reset and is programmable by the MPU. IER is also accessible at FE00D1. Bits 6-0 are undefined.

B.2.4 FE00C3: Transmit Interrupt (TIR - 1 Bit)

This bit (7) causes a level 4 autovectored interrupt request. The level 4 request will persist until the bit is cleared by software or a reset. TIR is also accessible at FE00D3. Bits 6-0 are undefined.

B.2.5 FE00C5: Receive Interrupt (RIR - 1 Bit)

This bit (7) causes a level 5 autovectored interrupt request. The level 5 request will persist until the bit is cleared by reset or software. RIR is also accessible at FE00D5. Bits 6-0 are undefined.

B.2.6 FE00C7: Utility Interrupt (UIR - 1 Bit)

This bit (7) causes a level 2 autovectored interrupt request. The level 2 request will persist until the bit is cleared by software or a reset. UIR is also accessible at FE00D7. Bits 6-0 are undefined.

B.2.7 FE00CF: Map Register (MAPR - 1 Bit)

The MAPR is a one bit control register (bit 7) for control of whether RAM (MAPR = 1) or ROM (MAPR = 0) appears in the first 4K-bytes of Microprocessor Unit (MPU) memory space.

This register can be cleared/set by the program and cleared by reset. Bits 6-0 are undefined. MAPR is also accessible at FE00DF.

B.2.8 FE00E1: Exception Register (EXR - 4 Bits)

The EXR contains 4-bits which indicate the various types of exceptions. All EXR bits and the TIMER bit are simultaneously cleared by any write to the EXR. The EXR is also accessible at odd locations between FE00E3 and FE00FF. Bit definitions for the EXR are given in Table B-2.

TABLE B-2. Exception Register Bit Definitions

BIT	RW	SIGNAL MNEMONIC	SIGNAL NAME AND DESCRIPTION
7-4	-	Unused	Undefined.
3	RC	ACLO	AC line voltage low. This bit is set when ACFAIL* is active on the VMEbus. It causes a level 7 autovector interrupt request.
2	RC	ABO	Abort. This bit is set when the external software ABORT switch is closed. It causes a level 7 autovector interrupt request.
1	RC	PER	Parity Error. Indicates a RAM parity error occurred on a read operation. It causes a level 7 autovector interrupt request.
0	RC	RTO	Resource Time-out. This bit indicates that the reason for a bus error condition was resource time-out. A resource time-out occurs if access to an on or offboard resource takes longer than 8 us (16 us - jumperable via JP04).

B.2.9 FE00C9, FE00CB, and FE00CD: Unused Registers (1 Bit)

The single bit (bit 7) read/write registers at FE00C9, FE00CB, and FE00CD are not assigned a specific use and may be used for signaling or future enhancements to the MVME330. Bits 6-0 are undefined. These registers may also be accessed at FE00D9, FE00DB, and FE00DD, respectively.

B.2.10 FE0200: LANCE Data Register (LDR - 4 Bits)

The LDR is a read/write word-only access port to the LANCE control/status registers CSR0, CSR1, CSR2, and CSR3. The specific meaning of bits and fields in programming this port are detailed in the AMD Am7990 Specification. This port is also accessible at addresses <FE0200 + 4*n> where 0<n<80x.

B.2.11 FE0202: LANCE Address Register (LAR - 2 Bits)

The LAR is used to select which of the four LANCE CSR's is to be accessed via the LANCE Data Register port. Only bits 0 and 1 have significance. Bits 15-2 read 0. The use of this register and CSR3-CSR0 are detailed in the AMD Am7990 specification. This port is also accessible at addresses <FE0202 + 4*n> where 0<n<80x.

B.2.12 FE0400: Network Parameters PROM (NPP - 512 X 4))

The Network Parameters PROM is a 512 word X 4 bit PROM accessible between FE0400 and FE07FE. Bits 3-0 of each word have significance while bits 15-2 are undefined. The contents of the NPP should include the station address assigned to the MVME330.

B.3 INTERRUPTS

The MVME330 receives interrupt requests from its internal as well as external resources. The interrupts are autovectored for hardware simplicity. The sources, priority, and type of interrupts are described in Table B-3 below.

TABLE B-3. Interrupts

LEVEL	NAME	FUNCTION
7A	TIMER	Real time clock; refresh time-out
7A	ACLO	Power failing
7A	ABO	Software ABORT switch
7A	PER	RAM parity error
6A	LANCE	LANCE interrupt
5A	RINT	Receive interrupt
4A	TINT	Transmit interrupt
3A	BINT	Host to MVME330 interrupt (self-clearing)
2A	UINT	Utility interrupt
1A	UNUSED	

NOTES: 1. "A" denotes Autovector.

2. All of the above interrupts except BINT require the MPU to clear the interrupting condition before RTE. All interrupts are enabled/disabled by writing a I/O to IEN (FE00C1, Bit 7).

APPENDIX C - MAIN MEMORY MAP

PHYSICAL ADDRESS RANGE (HEXADECIMAL)	DEVICES ACCESSED	PORT SIZE	SIZE (BYTES)
00000000 - 003FFFFF	Onboard DRAM	D32	4Mb
00400000 - 00EFFFFF	VMEbus A32/A24	D32/D16	11Mb
00F00000 - FFEFFFFF	VMEbus A32	D32/D16	4Gb
FFF00000 - FFF1FFFF	ROM/EEPROM bank 1	D16	128Kb
FFF20000 - FFF3FFFF	ROM/EEPROM bank 2	D16	128Kb
FFF40000 - FFF5FFFF	ROM/EEPROM bank 1 repeats in this space.	D16	128Kb
FFF60000 - FFF7FFFF	ROM/EEPROM bank 2 repeats in this space.	D16	128Kb
FFF80000 - FFF9FFFF	MSR & MC68901 MFP	D16	128Kb
FFFA0000 - FFFBFFFF	Z8530 Serial Communica-tions Controller (SCC)	D08	128Kb
FFFC0000 - FFFCFFFF	MK48T02 real-time clock with 2Kb SRAM	D08	64Kb
FFFD0000 - FFFDFFFF	PUPMMU Flag	--	64Kb
FFFE0000 - FFFEFFFF	VMEbus Interrupter	D08	64Kb
FFFF0000 - FFFFFFFF	VMEbus Short I/O Space	D08	64Kb